

# NAVAL POSTGRADUATE SCHOOL MONTEREY, CALIFORNIA



## THESIS

### SENSORS AND CONTROL IN THE UNDERWATER INTERCEPT AND DOCKING PROBLEMS

by

João P. C. Roque

June, 1995

Thesis Advisor:

Robert G. Hutchins

Approved for public release; distribution is unlimited.

19960122 130

DTIC QUALITY INSPECTED 1

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.				
1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE June 1995	3. REPORT TYPE AND DATES COVERED Master's Thesis		
4. TITLE AND SUBTITLE SENSORS AND CONTROL IN THE UNDERWATER INTERCEPT AND DOCKING PROBLEMS		5. FUNDING NUMBERS		
6. AUTHOR(S) Roque, João P. C.				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey CA 93943-5000		8. PERFORMING ORGANIZATION REPORT NUMBER		
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSORING/MONITORING AGENCY REPORT NUMBER		
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.				
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.		12b. DISTRIBUTION CODE		
13. ABSTRACT (maximum 200 words) Autonomous Underwater Vehicles (AUV) have been used to accomplish a variety of missions, and their roles are expanding as sensors, computers and algorithms became more sophisticated. Sensors and control algorithms are of primary importance in solving a target intercept or a docking problem. Three control algorithms (pursuit guidance, proportional navigation and a linear quadratic regulator) and three target information scenarios (perfect target information, noisy range and angle measurements and noisy angle measurements) are investigated here. This thesis presents the results of a simulation study for an Anti-Torpedo Torpedo (ATT) intercept problem for three different target trajectories, three target information scenarios and three guidance algorithms. This simulation study also includes the docking of an AUV to a moving platform. The proportional navigation algorithm is the most accurate for use in the ATT intercept problem, always achieving the smallest miss distances. However, it is of little use in the docking problem because it cannot control relative velocities. The linear quadratic regulator is successfully used to dock an AUV to a moving platform.				
14. SUBJECT TERMS Sensors, guidance, extended Kalman filter, intercept, docking, simulation		15. NUMBER OF PAGES 70		
		16. PRICE CODE		
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL	



Approved for public release; distribution is unlimited.

**SENSORS AND CONTROL IN THE UNDERWATER INTERCEPT AND  
DOCKING PROBLEMS**

João P. C. Roque  
Lieutenant Junior Grade, Portuguese Navy  
B.S., Portuguese Naval Academy, 1989

Submitted in partial fulfillment  
of the requirements for the degree of

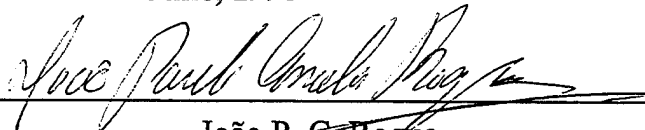
**MASTER OF SCIENCE IN ELECTRICAL ENGINEERING**

from the

**NAVAL POSTGRADUATE SCHOOL**

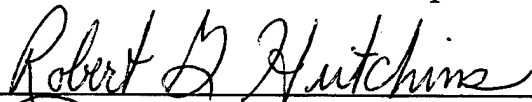
**June, 1995**

Author:

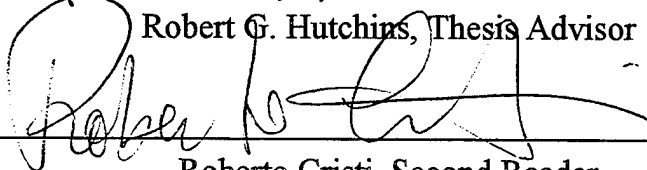


João P. C. Roque

Approved by:



Robert G. Hutchins, Thesis Advisor



Roberto Cristi, Second Reader



Michael A. Morgan, Chairman

Department of Electrical and Computer Engineering



## ABSTRACT

Autonomous Underwater Vehicles (AUV) have been used to accomplish a variety of missions, and their roles are expanding as sensors, computers and algorithms became more sophisticated. Sensors and control algorithms are of primary importance in solving a target intercept or a docking problem. Three control algorithms (pursuit guidance, proportional navigation and a linear quadratic regulator) and three target information scenarios (perfect target information, noisy range and angle measurements and noisy angle measurements) are investigated here. This thesis presents the results of a simulation study for an Anti-Torpedo Torpedo (ATT) intercept problem for three different target trajectories, three target information scenarios and three guidance algorithms. This simulation study also includes the docking of an AUV to a moving platform. The proportional navigation algorithm is the most accurate for use in the ATT intercept problem, always achieving the smallest miss distances. However, it is of little use in the docking problem because it cannot control relative velocities. The linear quadratic regulator is successfully used to dock an AUV to a moving platform.



## TABLE OF CONTENTS

I.	INTRODUCTION .....	1
A.	BACKGROUND .....	1
B.	OBJECTIVES .....	2
C.	THESIS OUTLINE .....	3
II.	MATHEMATICAL MODELS AND SIMULATION .....	5
A.	GENERAL .....	5
B.	COORDINATE SYSTEMS .....	5
C.	EQUATIONS OF MOTION .....	6
D.	SIMULATION .....	8
E.	KALMAN ESTIMATOR .....	9
III.	CONTROL ALGORITHMS .....	13
A.	GENERAL .....	13
B.	PURSUIT GUIDANCE .....	13
C.	PROPORTIONAL NAVIGATION .....	14
D.	LINEAR QUADRATIC REGULATOR .....	15
IV.	THE ATT INTERCEPT PROBLEM .....	19
A.	GENERAL .....	19
B.	ESTIMATOR PERFORMANCE .....	19
C.	ALGORITHMS PERFORMANCE .....	28
V.	THE DOCKING PROBLEM .....	35
A.	GENERAL .....	35
B.	DOCKING METHOD .....	35



VI. SUMMARY, CONCLUSIONS AND RECOMMENDATIONS .....	39
A. SUMMARY .....	39
B. CONCLUSIONS .....	39
C. RECOMMENDATIONS .....	40
APPENDIX. MATLAB SOURCE CODE .....	41
A. TRUE TARGET INFORMATION INTERCEPT .....	41
1. System Initialization .....	41
2. Simulation Diagram .....	42
3. Target Dynamics .....	42
4. ATT Dynamics .....	43
5. Pursuit Guidance .....	43
6. Proportional Navigation .....	44
7. Linear Quadratic Regulator .....	45
B. RANGE AND ANGLE MEASUREMENTS INTERCEPT .....	46
1. System Initialization .....	46
2. Simulation Diagram .....	47
3. Observations Delay .....	47
4. Noise Saturation .....	48
5. Extended Kalman Filter .....	48
C. ANGLE ONLY MEASUREMENTS INTERCEPT .....	49
1. System Initialization .....	49
2. Simulation Diagram .....	50
3. Kalman Filter .....	50
4. Pursuit Guidance .....	51
5. Proportional Navigation .....	51
D. DOCKING .....	52
1. System Initialization .....	52

2. Simulation Diagram .....	54
3. Docking Algorithm .....	54
LIST OF REFERENCES .....	57
INITIAL DISTRIBUTION LIST .....	59

## ACKNOWLEDGMENTS

I would like to thank my thesis advisor, Prof. Hutchins, for his guidance, availability and support, in all stages of the work.

I would like to thank my wife Silvia, for her support and understanding. To her I dedicate this thesis.

# I. INTRODUCTION

## A. BACKGROUND

Autonomous Underwater Vehicles (AUV) have been used for some time to accomplish a variety of missions. This thesis will approach two of these missions, the intercept problem for an Anti-Torpedo Torpedo (ATT) and the docking problem for a more general AUV. For these problems, the guidance algorithm used and the quality of the target estimates are major factors in mission success.

A general assumption made in the study of intercept weapons is that the interceptor has both speed and maneuverability advantages over the target. This is often true in a missile-versus-aircraft scenario but is false in the underwater ATT scenario. The simulation of the ATT intercept problem studied here will assume that both vehicles have similar speeds and similar maneuverability characteristics. Another limitation in the underwater ATT scenario is the target measurement delay caused by the sound propagation speed in salt water, a result of the almost universal use of acoustic sensors at intermediate and long ranges in the ocean, as compared to optical or electromagnetic measurements used in aerial intercepts. Acoustic underwater measurements are also less precise than electromagnetic ones. The nonlinearities of the conducting medium are more pronounced in salt water than in air, and sonars have, in general, less resolution than radars. To take all of this into account, the simulation process assumes the existence of large noise levels and time delays in the measurements.

The ATT sensors will generally be similar to torpedo acoustic sensors measuring the line-of-sight angles to the target. Two different target information scenarios will be considered, the case where the ATT has an active sensor providing both range and angular information, and the case where the ATT has only passive sensors providing only angular information. Two different Extended Kalman Filters will be designed to process the noisy measurements. Also, two different target maneuvering scenarios will be studied in detail: a target in straight line motion at a constant speed and a target performing a zigzag maneuver at constant speed. A target in a sinusoidal motion pattern will also be included for some comparisons.

The docking problem is inherently more difficult because it attempts to match both position and velocity coordinates of the docking vehicle with a fixed position (the dock) or (even more difficult) with a moving target. This thesis will examine the more difficult problem.

The docking of vehicles is generally done in "manual mode". To perform a docking maneuver in "automatic mode" raises a series of problems that do not exist in the intercept problem. Most of the known guidance algorithms for use in weapons are designed only to minimize the position difference between the commanded vehicle and the target vehicle. For the docking problem there must also be a velocity and/or orientation match between the two vehicles or between the vehicle and the fixed dock. Because this is a very general problem, this thesis will try to solve one of the harder cases, the docking of an autonomous vehicle to a moving target. The simulation will also assume that the AUV has maneuverability limitations but has, for this specific case, a speed advantage over the target. Only with a speed advantage can the docking be accomplished, allowing the AUV to close on the target from the rear.

## **B. OBJECTIVES**

The objectives of this thesis are to investigate the performance of control algorithms for the intercept and for the docking problems. The evaluation of control algorithms will be based on the miss distance between the ATT and the target. Different simulations will be made to find the influence of individual factors on the performance of each guidance law. The Kalman filter estimates will also be analyzed, and the feasibility of using the Kalman filter to process noisy angle-only measurements will be studied. In the docking problem, the objective is to derive a control algorithm that will work for arbitrary initial positions and velocities.

The simulation will be done using MATLAB with SIMULINK. The simulation program is divided into a number of blocks performing very specific tasks. This way it will be easy to modify the simulation just by replacing individual M-Files or S-functions.

### **C. THESIS OUTLINE**

The analysis, as well as the results obtained from simulations, will be presented in the five subsequent chapters. Chapter II describes the vehicle's equations of motion, a simulation overview and the two versions of Extended Kalman filter. In Chapter III, the control algorithms used in the simulation are presented. Chapter IV describes the setup for the intercept problem and presents the simulation results, and Chapter V does a similar job for the docking problem. Chapter VI summarizes the results and conclusions from the evaluation of both problems. Finally, the Matlab M-files and S-functions are presented in the Appendix.



## II. MATHEMATICAL MODELS AND SIMULATION

### A. GENERAL

This chapter describes the coordinate systems and the mathematical models used for the underwater vehicles, the measurement model for the sensors used in the AUV and the extended Kalman filter estimator.

### B. COORDINATE SYSTEMS

The motion of vehicles is generally described using two different reference frames, an inertial reference frame in global coordinates, Figure 1, and a body fixed coordinate system with the vehicle's velocity vector aligned with the  $x$ -axis and orthogonal  $y$  and  $z$  axes corresponding to yaw and pitch planes, Figure 2. This simulation will use a total of three reference frames: an inertial one that has its origin at the launch position of the Autonomous Underwater Vehicle, and two others with body-fixed coordinate axes aligned at all times with the individual vehicles, AUV and target. For the angular measures to be expressed with standard compass angles, the inertial reference axis has the  $x$ -axis pointing toward North, the  $y$ -axis pointing toward East and the  $z$ -axis pointing toward the zenith. Each body-fixed

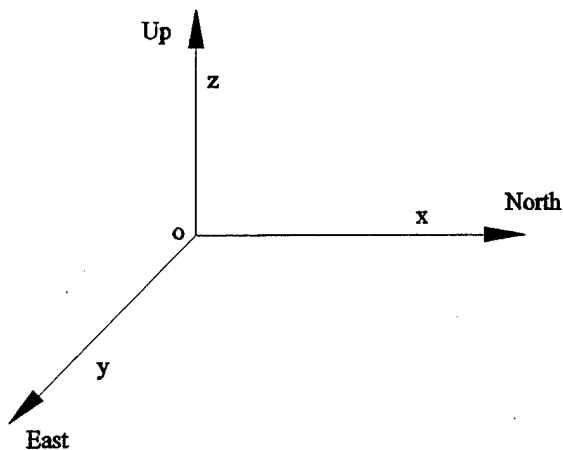


Figure 1. Inertial reference frame.

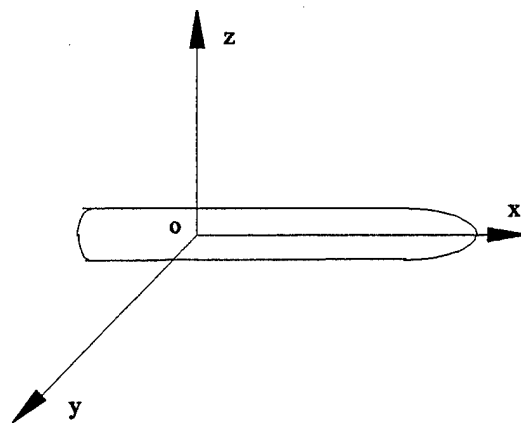


Figure 2. Body-fixed reference frame.



coordinate frame has its  $x$ -axis aligned with the vehicle's longitudinal axis of symmetry, which is assumed to coincide with the direction of the vehicle's velocity vector.

Position of the body-fixed axis is expressed by the inertial position of its origin, and its orientation is defined by the Euler angles  $\Psi$ ,  $\Theta$  and  $\Phi$ . Here  $\Psi$  is the angle between inertial and body-fixed  $x$ -axes measured in the horizontal plane,  $\Theta$  is the angle between the horizontal and the body-fixed  $x$ -axes measured in the vertical plane, and  $\Phi$  is the angle between the horizontal and the body-fixed  $y$ -axes measured in the  $OYZ$  body-fixed plane.

### C. EQUATIONS OF MOTION

The equations of motion used in the simulation are a set of general and nonlinear differential equations that relate the vehicle's velocities in body-fixed coordinates ( $u, v, w$ ) and the Euler angles ( $\Psi$ ,  $\Theta$ ,  $\Phi$ ) with the vehicle's velocities expressed in inertial coordinates ( $\dot{x}=v_x$ ,  $\dot{y}=v_y$ ,  $\dot{z}=v_z$ ):

$$\begin{aligned}\dot{x} &= u \cos \Psi \cos \Theta \\ &+ v (\cos \Psi \sin \Theta \sin \Phi - \sin \Psi \cos \Phi) \\ &+ w (\cos \Psi \sin \Theta \cos \Phi + \sin \Psi \sin \Phi) \\ \dot{y} &= u \sin \Psi \cos \Theta \\ &+ v (\sin \Psi \sin \Theta \sin \Phi + \cos \Psi \cos \Phi) \\ &+ w (\sin \Psi \sin \Theta \cos \Phi - \cos \Psi \sin \Phi) \\ \dot{z} &= u \sin \Theta \\ &+ v \cos \Theta \sin \Psi \\ &+ w \cos \Theta \cos \Psi,\end{aligned}\tag{2.1}$$

where  $u$  (surge),  $v$  (sway) and  $w$  (heave) are the vehicle's velocities in body-centered coordinates.  $\dot{x}$ ,  $\dot{y}$ , and  $\dot{z}$  are the vehicle's velocities in inertial coordinates. The Euler angular rates are functions of the preexisting Euler angles ( $\Psi$ ,  $\Theta$ ,  $\Phi$ ) and the vehicle turn rates in body-fixed coordinates ( $p, q, r$ ):

$$\begin{aligned}\dot{\Phi} &= p + q \sin \Phi \tan \Theta + r \cos \Phi \tan \Theta \\ \dot{\Theta} &= q \cos \Phi - r \sin \Phi \\ \dot{\Psi} &= q \frac{\sin \Phi}{\cos \Theta} + r \frac{\cos \Phi}{\cos \Theta},\end{aligned}\tag{2.2}$$

where  $p$ ,  $q$  and  $r$  are the respective Roll, Pitch and Yaw rates. All the vehicles' trajectories are simulated using equations 2.1 and 2.2. Vehicle velocities ( $u(t)$ ,  $v(t)$ ,  $w(t)$ ) and turn rates ( $p(t)$ ,

$q(t)$ ,  $r(t)$  are the results of control inputs to propellers, thrusters and/or steering planes. These control mechanisms and their dynamics are peculiar to each individual vehicle. In this study we assume  $v(t) = w(t) = 0$ . For the ATT study  $u(t)$  is taken as a constant, corresponding to a constant speed vehicle. For the docking problem  $u(t)$  is assumed to be subject to linear drag when adjusting to the desired control input velocity:  $\dot{u}(t) = -k u(t) + u_d$  where  $u_d$  is the desired control input velocity. Other control inputs are the turn rates  $p$ ,  $q$  and  $r$ . This model accurately simulates a vehicle such as a torpedo with an adjustable speed propeller or thruster system along the longitudinal body axis and control surfaces (steering planes) with negligible time constants for inducing turns. These assumptions are adequate for the proof of concept studies featured in this thesis, but more detail in modeling thrust and control surface dynamics would be required for a real vehicle simulation.

#### D. SIMULATION

Figure 3 represents a simplified block diagram of the code used in the simulation studies.

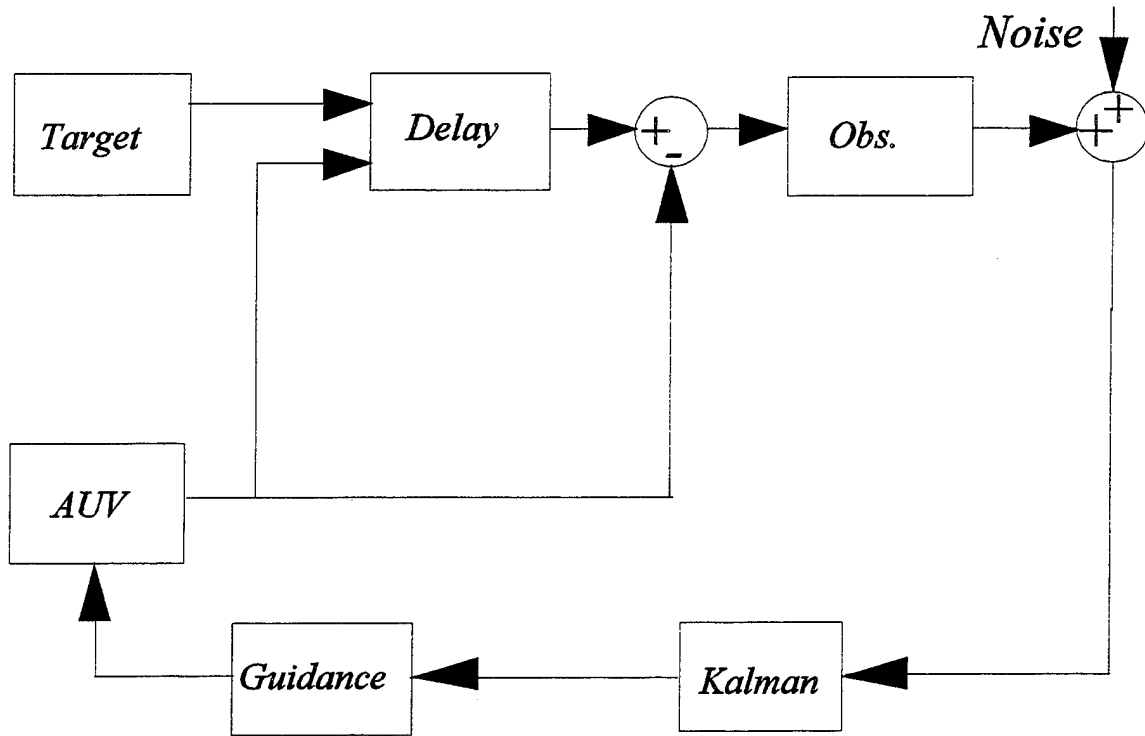


Figure 3. Simulation Diagram

The Target block simulates the motion of a target vehicle and outputs position and target velocity (given in inertial coordinates) in the six dimensional target state,  $\vec{x}_T$ .

The Delay block takes into account the sound propagation speed to delay the target information arrival to own sensors. This delay is calculated using the distance between vehicles and the sound propagation speed in salt water.

The Observation block receives the difference between AUV position and target delayed position and produces two angles, bearing and elevation, and a distance. The relative coordinates  $\vec{x}_r = \vec{x}_t - \vec{x}_a = [x \ y \ z]^T$  are transformed into spherical observations by

$$\begin{aligned} r &= \sqrt{x^2 + y^2 + z^2} \\ \alpha &= \arctan \frac{y}{x} \\ \beta &= \arctan \frac{z}{\sqrt{x^2 + y^2}} \end{aligned} \quad (2.3)$$

Simulations employing passive sensors do not use the range information.

After the measurements are produced, white noise is added to the observations. The noise variance is the same for both angle measurements. The AUV simulations use measurements with the following noise characteristics:

$$\sigma_r = 10.0 \text{ meters} \quad ; \quad \sigma_\alpha = \sigma_\beta = 1.0 \text{ degree} \quad .$$

The Kalman estimator, which is described below, estimates all the target states from the noisy observations. Two different Kalman filters were designed because of the use of range and bearing measurements in some scenarios and angle only measurements in others. The outputs of this block are the estimated target states relative to the AUV.

The Guidance block calculates the required control inputs to the AUV. These control inputs are all given in the body-fixed coordinate system.

The AUV block simulates the motion of the AUV given the control inputs. The outputs are the position and velocity of the AUV in inertial coordinates, given in the six dimensional state vector  $\vec{x}_a$ .

## E. KALMAN ESTIMATOR

The state differential equation for the target in the inertial reference system can be represented by the following linear model:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{v}_x \\ \dot{v}_y \\ \dot{v}_z \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ v_x \\ v_y \\ v_z \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix}. \quad (2.4)$$

Therefore, for a non-maneuvering target, the linear system is given by

$$\bar{x}((k+1)T_s) = \Phi(T_s)\bar{x}(kT_s) + \bar{w}_k \quad (2.5)$$

$$\bar{y}(kT_s) = C\bar{x}(kT_s) + \bar{v}_k, \quad (2.6)$$

with  $w_k$  and  $v_k$  the plant and measurement noise respectively. The term  $w_k$  also takes into account the fact that the target trajectory may not be linear. The state transition matrix,  $\Phi$ , is then

$$\Phi = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \quad (2.7)$$

The real measurements  $(r, \alpha, \beta)$  are assumed to be unbiased with covariance matrix

$$R^* = \begin{bmatrix} \sigma_r^2 & 0 & 0 \\ 0 & \sigma_\alpha^2 & 0 \\ 0 & 0 & \sigma_\beta^2 \end{bmatrix}, \quad (2.8)$$

where  $\sigma_r, \sigma_\alpha$  and  $\sigma_\beta$  are the standard deviations of the measurements. The measurements are preprocessed to give the pseudo measurements  $(x, y, z)$  with covariance matrix

$$R_k = G_k R^* G_k^T, \quad (2.9)$$

and

$$G_k = \begin{bmatrix} \frac{\partial g_1}{\partial r} & \frac{\partial g_1}{\partial \alpha} & \frac{\partial g_1}{\partial \beta} \\ \frac{\partial g_2}{\partial r} & \frac{\partial g_2}{\partial \alpha} & \frac{\partial g_2}{\partial \beta} \\ \frac{\partial g_3}{\partial r} & \frac{\partial g_3}{\partial \alpha} & \frac{\partial g_3}{\partial \beta} \end{bmatrix}, \quad (2.10)$$

evaluated at  $r(kT_s)$ ,  $\alpha(kT_s)$ , and  $\beta(kT_s)$  where

$$g(r, \alpha, \beta) = \begin{bmatrix} \sqrt{\frac{r^2}{a^2 b^2 + a^2 + b^2 + 1}} \\ \sqrt{\frac{r^2 a^2}{a^2 b^2 + a^2 + b^2 + 1}} \\ \sqrt{\frac{r^2 b^2}{1 + b^2}} \end{bmatrix}, \quad (2.11)$$

with  $a = \tan \alpha$  and  $b = \tan \beta$ .

With the preprocessed measurements, the C matrix is given by

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}. \quad (2.12)$$

For this problem, the discrete covariance matrix is

$$Q = \begin{bmatrix} \Sigma & 0 & 0 \\ 0 & \Sigma & 0 \\ 0 & 0 & \Sigma \end{bmatrix}_{6 \times 6}, \quad (2.13)$$

where

$$\Sigma = q^2 \begin{bmatrix} \frac{T_s^3}{3} & \frac{T_s^2}{2} \\ \frac{T_s^2}{2} & T_s \end{bmatrix}, \quad (2.14)$$

and  $q$  is a variable parameter that weights the relative importance between the assumed linear model and the noisy observations. This parameter is used to model the ability of the target to

maneuver. High values of  $q$  will enable the Kalman filter to follow the target during turns, while low values of  $q$  produce a smooth estimate.

The discrete time Kalman filter can then be summarized by a prediction step:

$$\hat{\mathbf{x}}_{k+1|k} = \Phi \hat{\mathbf{x}}_{k|k} \quad (2.15)$$

$$P_{k+1|k} = \Phi P_{k|k} \Phi^T + Q, \quad (2.16)$$

where  $P$  is the state error covariance matrix, and an update across a measurement step:

$$K_k = P_{k+1|k} C^T [C P_{k+1|k} C^T + R]^{-1} \quad (2.17)$$

$$\hat{\mathbf{x}}_{k+1|k+1} = \hat{\mathbf{x}}_{k+1|k} + K_k [y(kT_s) - C \hat{\mathbf{x}}_{k+1|k}] \quad (2.18)$$

$$P_{k+1|k+1} = [I - K_k C] P_{k+1|k} [I - K_k C]^T + K_k R K_k^T. \quad (2.19)$$

The above Kalman filter description assumes the existence of range and angle measurements. When the only observations are angles, the target system is not totally observable. The target model was modified, and now the system states are angles and angular rates.

$$\begin{bmatrix} \dot{\alpha} \\ \dot{\beta} \\ \dot{\omega}_\alpha \\ \dot{\omega}_\beta \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ \omega_\alpha \\ \omega_\beta \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_\alpha \\ u_\beta \end{bmatrix}. \quad (2.20)$$

The estimate is now restricted to angles and angular rates, which limits the number of control algorithms that can be used. However, the estimation process is similar to the one described above.

### III. CONTROL ALGORITHMS

#### A. GENERAL

This chapter describes the three control laws used in the simulation, Pursuit Guidance, Proportional Navigation and a Linear Quadratic Regulator. The first two laws are designed to achieve a target interception, while the third one can be used in a more general way, as for example, to dock a vehicle to a moving platform. The control inputs are calculated as desired vehicle turn rates. Desired speed is also a control input for the docking problem.

#### B. PURSUIT GUIDANCE

The purpose of this guidance law is to point the AUV velocity vector at the target. The control inputs are calculated using the angular differences between the line-of-sight vector to the target and the AUV velocity vector.

The relative target positions  $\vec{r}_r = \vec{r}_t - \vec{r}_a$  are rotated to the AUV body fixed coordinate system by

$$\vec{r}_{r|body} = \begin{bmatrix} \cos(\Psi_a)\cos(\Theta_a) & \sin(\Psi_a)\cos(\Theta_a) & -\sin(\Theta_a) \\ -\sin(\Psi_a) & \cos(\Psi_a) & 0 \\ \cos(\Psi_a)\sin(\Theta_a) & \sin(\Psi_a)\sin(\Theta_a) & \cos(\Theta_a) \end{bmatrix} \vec{r}_{r|inertial}, \quad (3.1)$$

where

$$\Psi_a = \arctan \frac{v_{y_a}}{v_{x_a}} \quad (3.2)$$

$$\Theta_a = \arctan -\frac{v_{z_a}}{\sqrt{v_{x_a}^2 + v_{y_a}^2}}, \quad (3.3)$$

and  $\vec{r}_{r|body} = [x_{rb} \ y_{rb} \ z_{rb}]^T$ .

The target angles in the body centered horizontal and vertical planes are then



$$\alpha_y = \arctan \frac{y_{rb}}{\sqrt{x_{rb}^2 + y_{rb}^2 + z_{rb}^2}} \quad (3.4)$$

$$\alpha_z = \arctan \frac{z_{rb}}{\sqrt{x_{rb}^2 + y_{rb}^2 + z_{rb}^2}} \quad (3.5)$$

The resulting inputs are yaw and pitch rates,  $u_y$  and  $u_z$ , given by

$$\begin{aligned} u_y &= k_p \alpha_y \\ u_z &= k_p \alpha_z, \end{aligned} \quad (3.6)$$

where  $k_p$  is a pursuit guidance constant. A value of  $k_p=3$  is used throughout the simulation.

When the total calculated acceleration exceeds the maximum allowable input acceleration, they are reduced using the following equations:

$$\begin{aligned} \text{if } u &= \sqrt{u_y^2 + u_z^2} > u_{\max} \\ u_y &= u_y \frac{u_{\max}}{u} \\ u_z &= u_z \frac{u_{\max}}{u}. \end{aligned} \quad (3.7)$$

### C. PROPORTIONAL NAVIGATION

This control algorithm drives the AUV into a collision course with the target. This is accomplished when the rate of change of the line-of-sight vector is zero.

Relative target position and velocity are rotated to the body centered coordinate system as in the previous example. The line-of-sight rate is defined by the cross product between the relative position and velocity vectors, given in AUV body-fixed coordinates.

$$\bar{\omega}_b = \frac{\bar{r}_{r|body} \times \bar{v}_{r|body}}{|\bar{r}_{r|body}|^2} \quad (3.8)$$

With  $\bar{e}_x = [1 \ 0 \ 0]^T$ , the unit vector along the AUV velocity vector, the required direction for the control inputs is given by

$$\begin{bmatrix} 0 \\ \cos(\gamma) \\ \sin(\gamma) \end{bmatrix} = \frac{\vec{w}_b \times \vec{e}_x}{|\vec{w}_b \times \vec{e}_x|}. \quad (3.9)$$

The resulting control inputs are then

$$\begin{aligned} u_y &= \frac{k_{pn} \cos(\gamma) |\vec{v}_b| |\vec{w}_b|}{|\vec{v}_a|} \\ u_z &= \frac{k_{pn} \sin(\gamma) |\vec{v}_b| |\vec{w}_b|}{|\vec{v}_a|}, \end{aligned} \quad (3.10)$$

where  $k_{pn}$  is a proportional navigation constant set equal to 4 throughout the simulation,  $\vec{v}_b$  is the relative velocity vector calculated above, which represents the closing speed between the two underwater vehicles, and  $\vec{v}_a$  is the AUV speed.

If the total command input exceeds the maximum allowable one,  $u_y$  and  $u_z$  are reduced as before.

#### D. LINEAR QUADRATIC REGULATOR

To apply an Optimal Linear Control the plant equations need to be linearized. Using the relative positions in the body centered coordinate system and assuming an AUV constant speed, the system can be represented by

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{v}_x \\ \dot{v}_y \\ \dot{v}_z \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ v_x \\ v_y \\ v_z \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_y \\ u_z \end{bmatrix}. \quad (3.11)$$

However, this linearization of the real system has degenerated into an uncontrollable system. To fix this, equations permitting a variable thrust input were employed. Hence, the following set of linearized state equations was used.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{v}_x \\ \dot{v}_y \\ \dot{v}_z \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ v_x \\ v_y \\ v_z \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix}, \quad (3.12)$$

where  $u_x$  is the surge rate,  $u_y$  is the centripetal acceleration in the  $XY$  plane given by vehicle speed times yaw rate, and  $u_z$  is the corresponding acceleration in the  $XZ$  plane given by vehicle speed times pitch rate. For the ATT intercept problem,  $u_x$  is zero.

The objective of this control algorithm is to minimize a cost function defined by

$$J = \int_{t_1}^{t_2} [\bar{x}(t)^T Q \bar{x}(t) + \bar{u}(t)^T R \bar{u}(t)] dt, \quad (3.13)$$

where  $Q$  and  $R$  are weighting matrices.  $Q$  is a symmetric and non-negative definite matrix that weights the relative importance of the states' deviation from zero.  $R$  is a symmetric, positive definite matrix that limits the control energy during the trajectory.

Control inputs are then given by

$$\bar{u}(t) = -F(t) \bar{x}(t), \quad (3.14)$$

where  $F(t)$  is the optimal feedback gain

$$F(t) = R^{-1}(t) B^T(t) K(t), \quad (3.15)$$

and, in steady state,  $K(t)$  satisfies a matrix Ricatti equation.

$$0 = \dot{K}(t) + Q(t) - K(t) B(t) R^{-1}(t) B^T(t) K(t) + K(t) A(t) + A^T(t) K(t). \quad (3.16)$$

This calculation is done using the discrete time equivalent system where the steady state  $F$  matrix is given by the MATLAB call,  $F = dlqr(\Phi, \Delta, Q, R)$ , with the system now characterized by a discrete time state equation

$$\bar{x}((k+1)T_s) = \Phi(T_s) \bar{x}(kT_s) + \Delta(T_s) \bar{u}(kT_s). \quad (3.17)$$

The choice of the weighting matrices  $Q$  and  $R$  is done according to the specific situation requirements.



## IV. THE ATT INTERCEPT PROBLEM

### A. GENERAL

This chapter describes the simulation of the intercept problem, as well as the results of the evaluation of the estimation filters and control algorithms. The case studied was an Anti-Torpedo Torpedo intercept problem. Three target motion scenarios were used to evaluate the guidance algorithms, a straight line, a zigzag pattern and a sinusoidal pattern.

### B. ESTIMATOR PERFORMANCE

Since the true target information is not available, an observer is needed to estimate the true target position and velocity or the true angles and angular rates. Gaussian white noise with  $\sigma_r = 10.0 \text{ meters}$  ;  $\sigma_\alpha = \sigma_\beta = 1.0^\circ$  is added to the observations. Since the sensors are placed in the interceptor vehicle, the range error is limited to 10% of the actual distance between target and interceptor for ranges less than 100 meters. To improve the filter performance at reduced range, the measurement covariance matrix used in the filter is adjusted for a range error  $\sigma_r = 1.0 \text{ meters}$  when the estimated range is less than 10 meters. The Kalman filter for angle only measurements suffers no alterations throughout the simulation.

Figures 4, 5 and 6 show examples of the estimated and true positions in the  $XY$  plane for each simulated target trajectory. These plots depict the typical estimation problems encountered when observing maneuvering targets. When the trajectory is not linear the estimation error increases considerably. As long as there is a long enough linear pattern after each turn, the estimates will converge to the true positions. The estimates of the sinusoidal target trajectory were not used as input for any guidance algorithm because of the large estimation errors. In that case the position errors were large and the velocity errors even larger, making the estimates of little use for any guidance law. Figures 7, 8, and 9 show the difference between estimated and real target positions, while Figures 10, 11 and 12 show the same difference for the velocities.

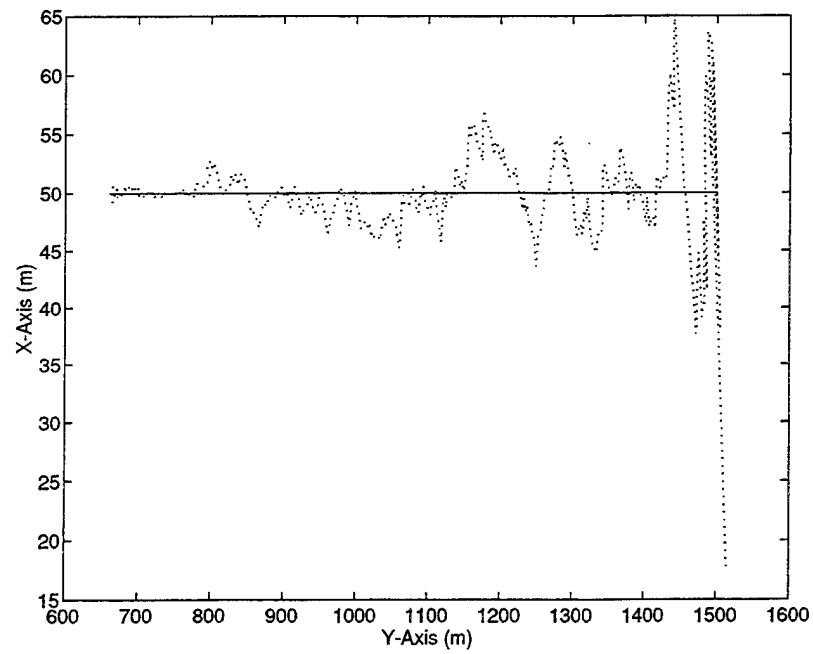


Figure 4. Actual and estimated target trajectory with no target maneuver.

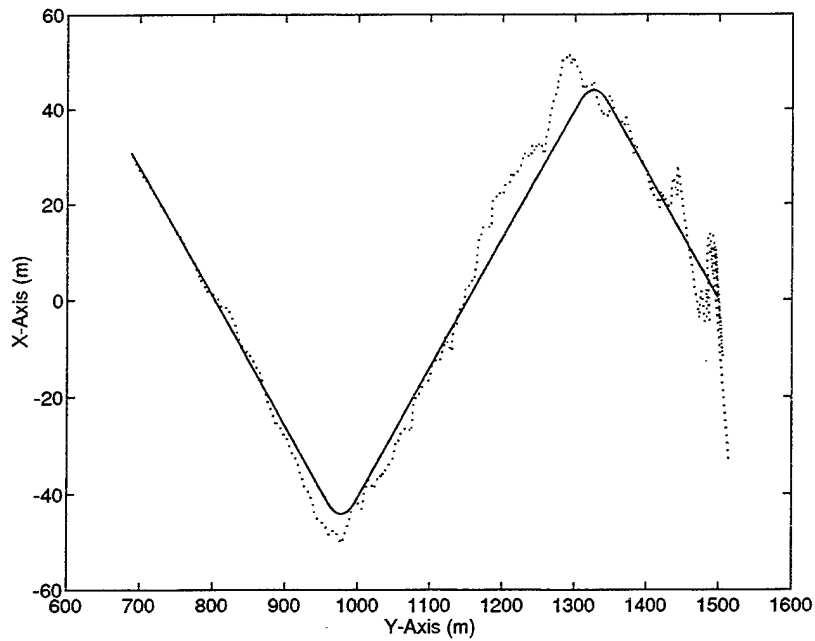


Figure 5. Actual and estimated target trajectory with zigzag target maneuver.

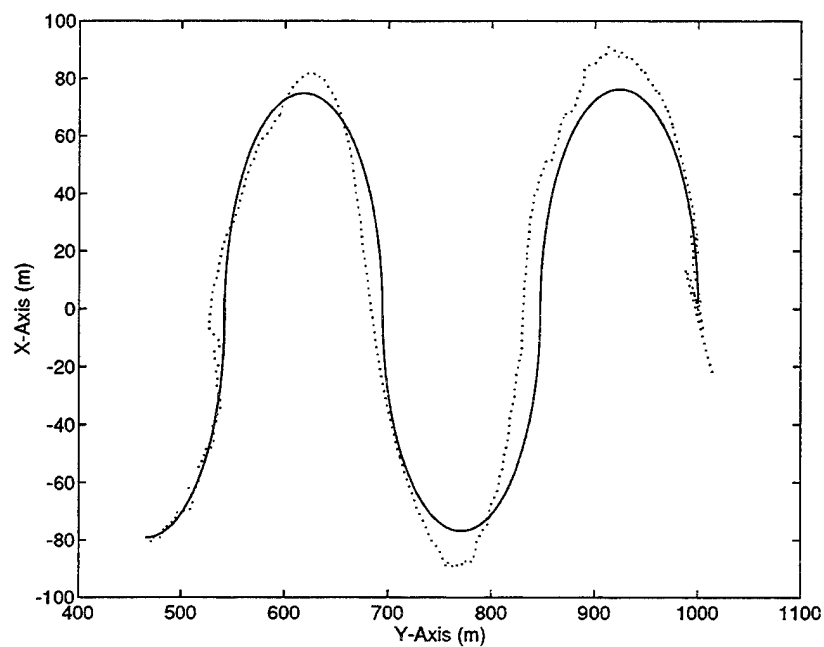


Figure 6. Actual and estimated target trajectory with sinusoidal target maneuver.

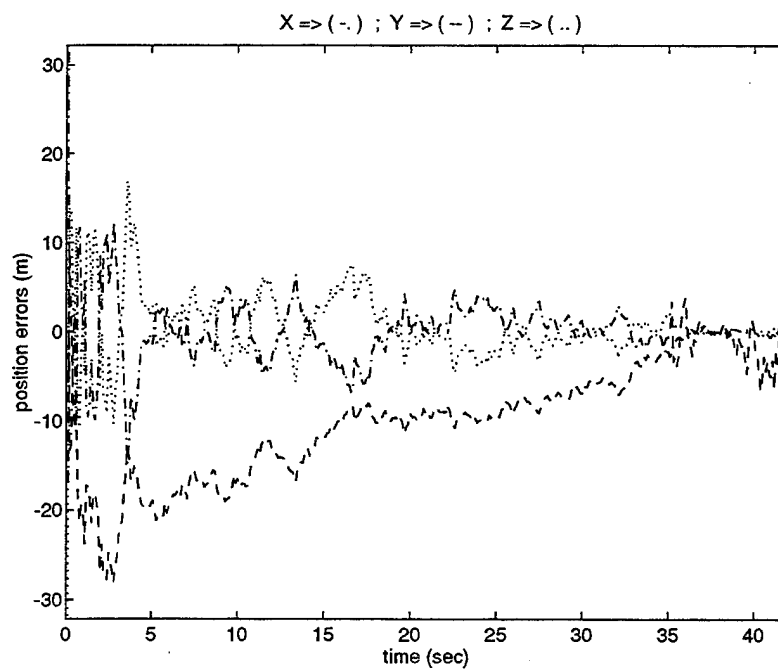


Figure 7. Errors in position estimates with no target maneuver.



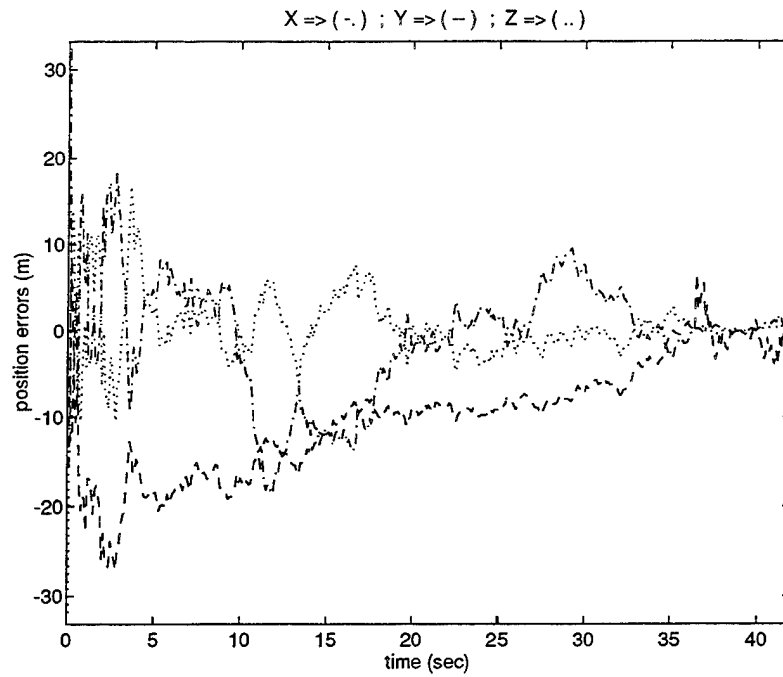


Figure 8. Errors in position estimates with zigzag target maneuver.

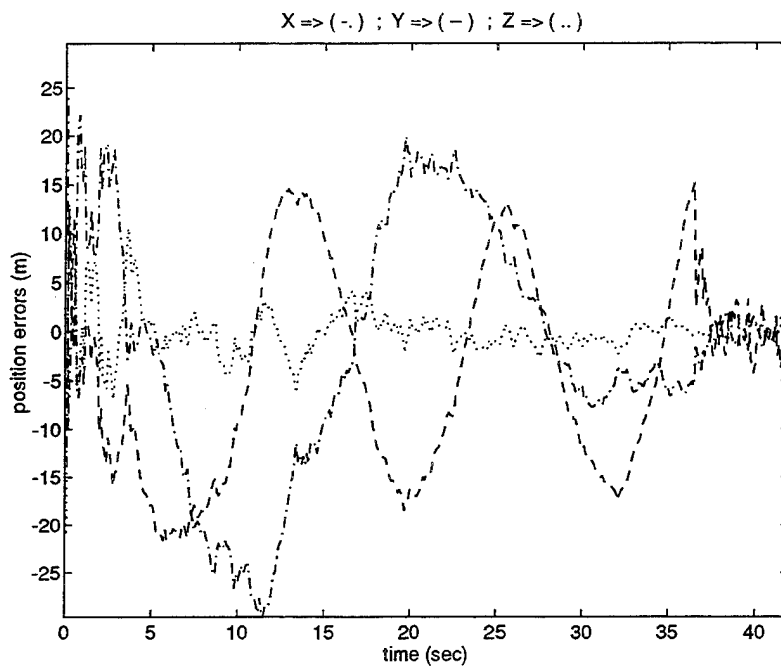


Figure 9. Errors in position estimates with sinusoidal target maneuver.

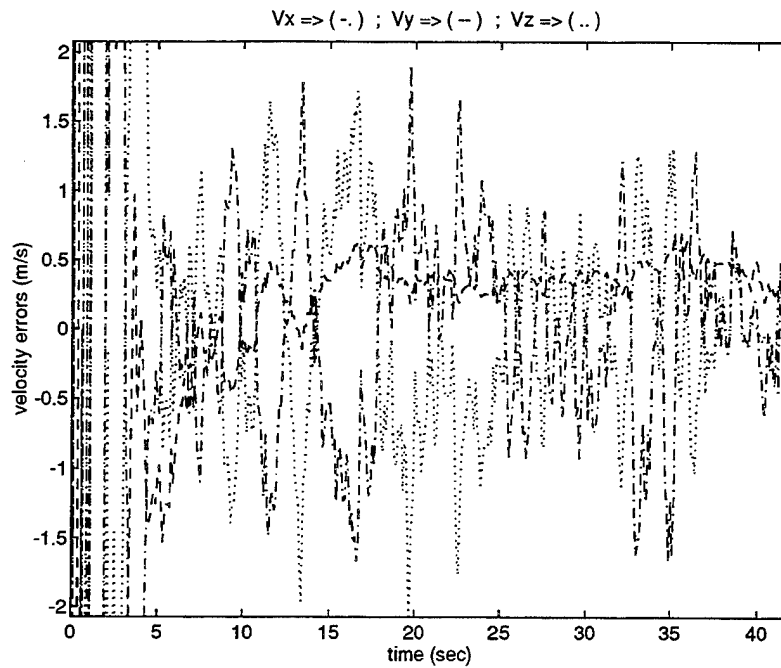


Figure 10. Errors in velocity estimates with no target maneuver.

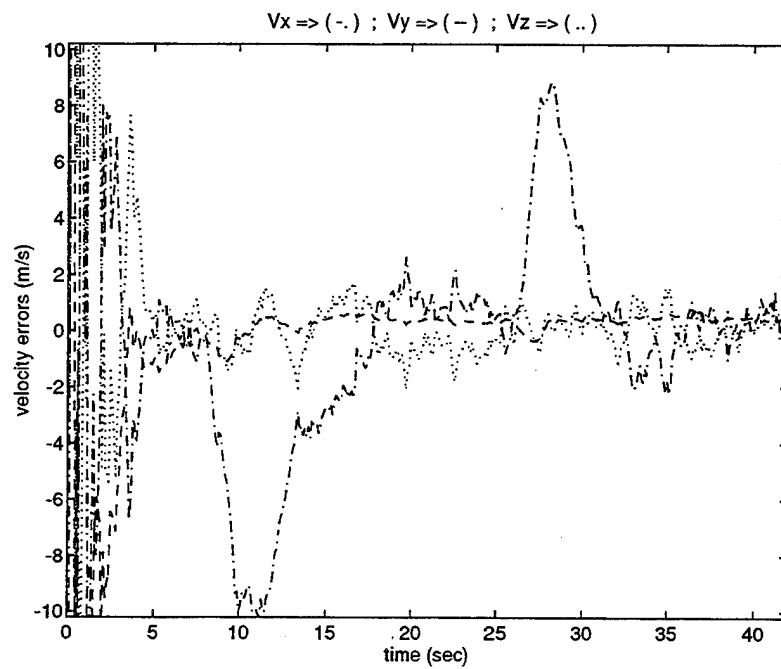


Figure 11. Errors in velocity estimates with zigzag target maneuver.

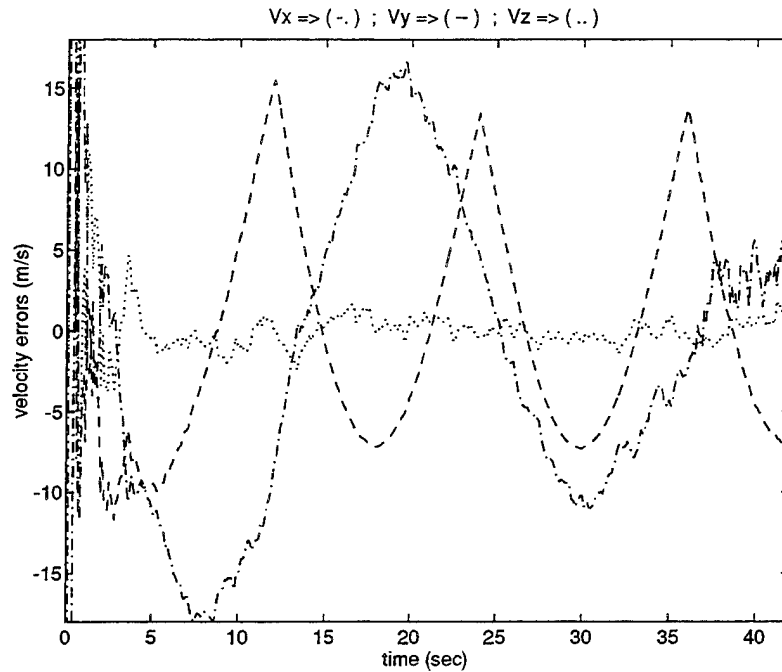


Figure 12. Errors in velocity estimates with sinusoidal target maneuver.

When only angle measurements are used, the Kalman filter is modified to estimate angle and angular rates. Figures 13, 14, and 15 show the actual and estimated line-of-sight angles while Figures 16, 17 and 18 show the actual and estimated angular rates for all three target trajectories. Because of the difficulty of estimating those parameters with an assumed linear model for a very nonlinear system and angle only observations, the simulation assumes that the estimator has the ability of producing numerical derivatives of the noisy angle measurements. Although this technique introduces very noisy pseudo angular rate measurements, the filter performance is, in general, improved.

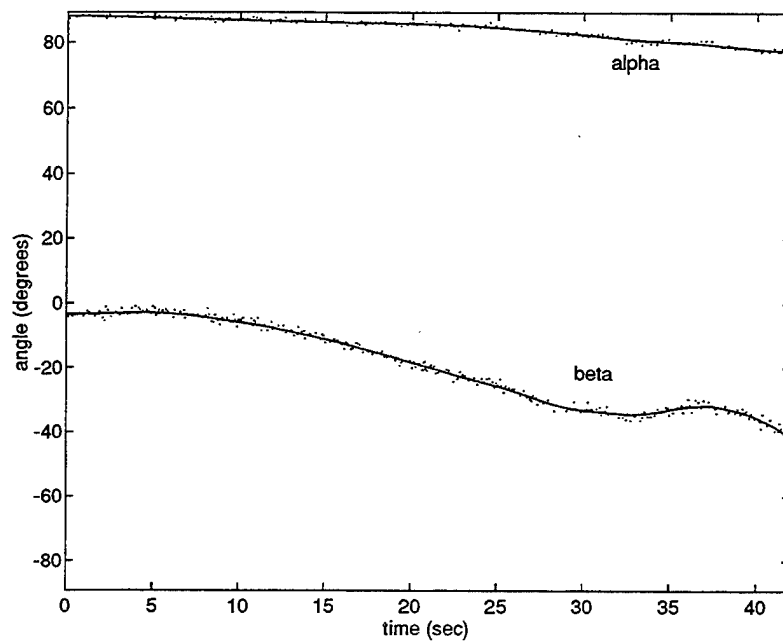


Figure 13. Actual and estimated line-of-sight angles with no target maneuver.

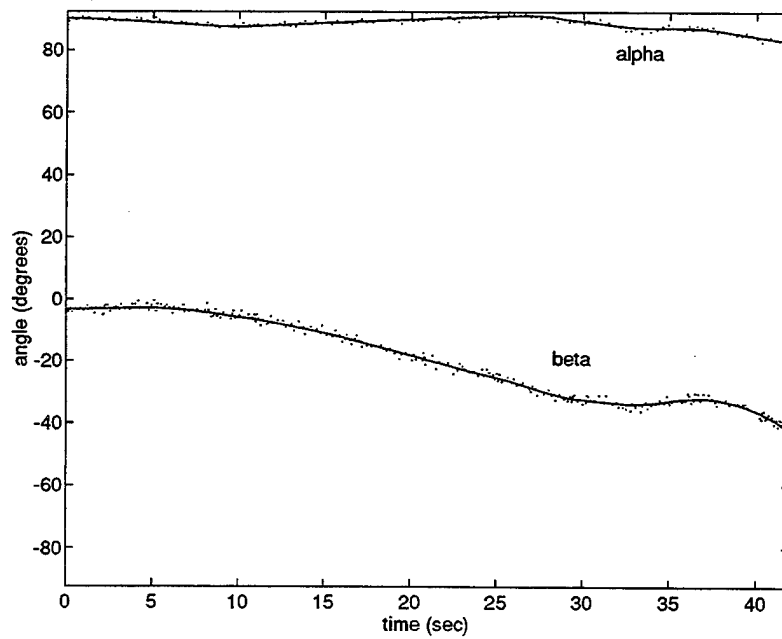


Figure 14. Actual and estimated line-of-sight angles with zigzag target maneuver.

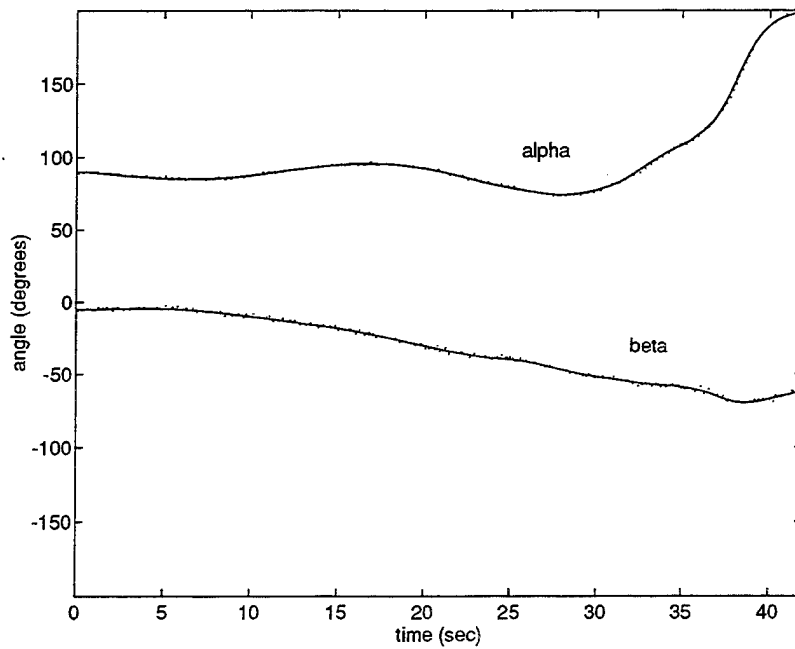


Figure 15. Actual and estimated line-of-sight angles with sinusoidal target maneuver.

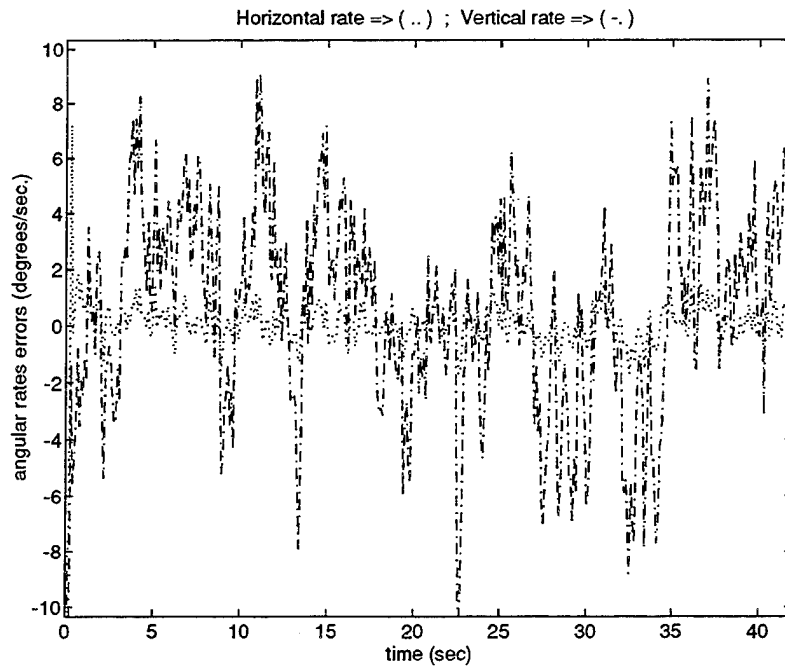


Figure 16. Errors in line-of-sight rate estimates with no target maneuver.

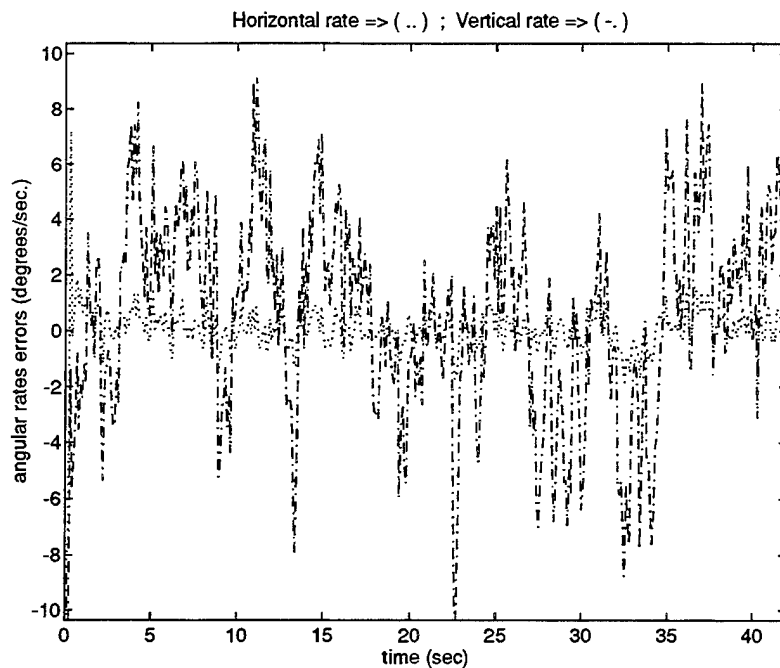


Figure 17. Errors in line-of-sight rate estimates with zigzag target maneuver.

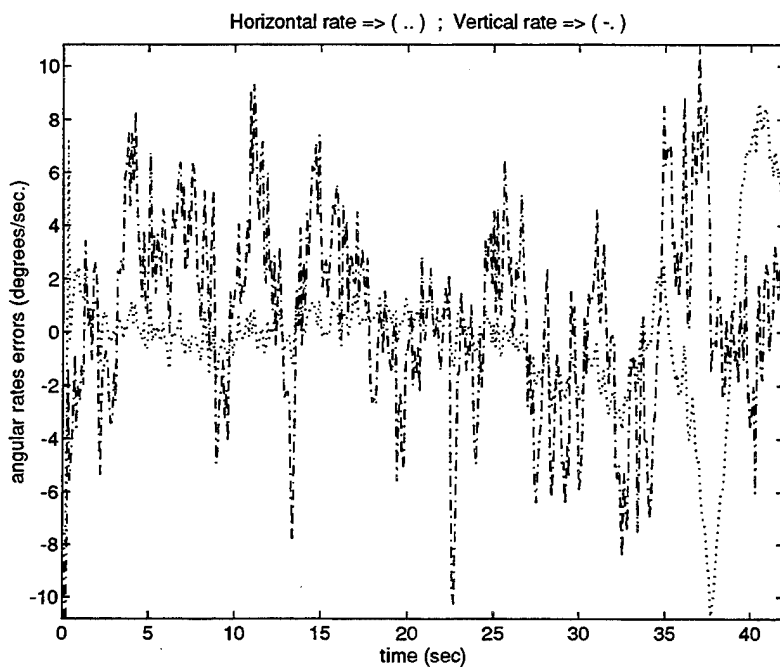


Figure 18. Errors in line-of-sight rate estimates with sinusoidal target maneuver.

### C. ALGORITHMS PERFORMANCE

To evaluate the control laws by themselves, independently of estimation errors, the simulation was also run with perfect target information. Perfect target information assumes that there is no delay in the target information. However, the delay due to the low propagation speed of sound proved to be of negligible significance. The sensors are placed on the ATT and, as the range difference between the vehicles decreases so does the information delay, ending up being negligible for short distances. Figures 19 through 27 show the trajectories of target and ATT for the three target trajectories and the three control laws. Also shown is the minimum miss distance for each case.

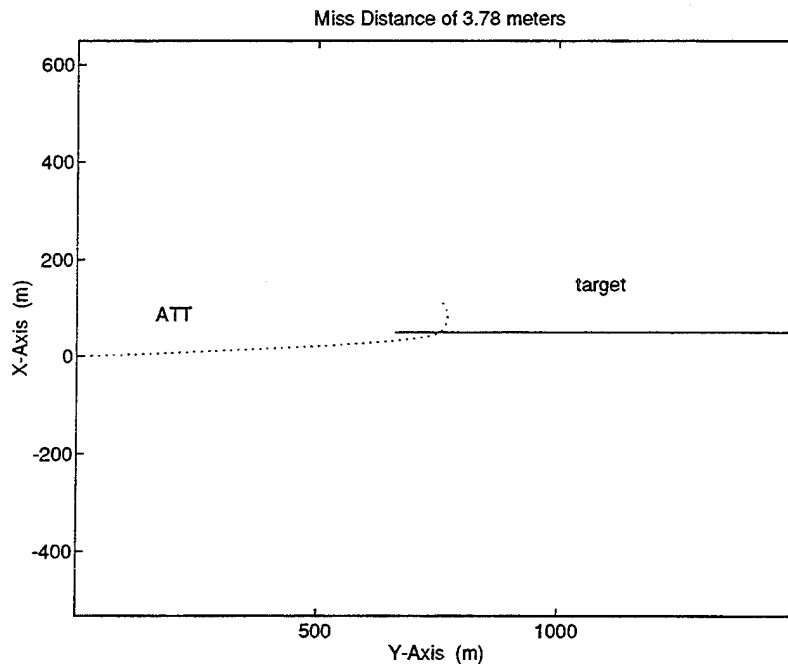


Figure 19. ATT and target trajectories with pursuit guidance.

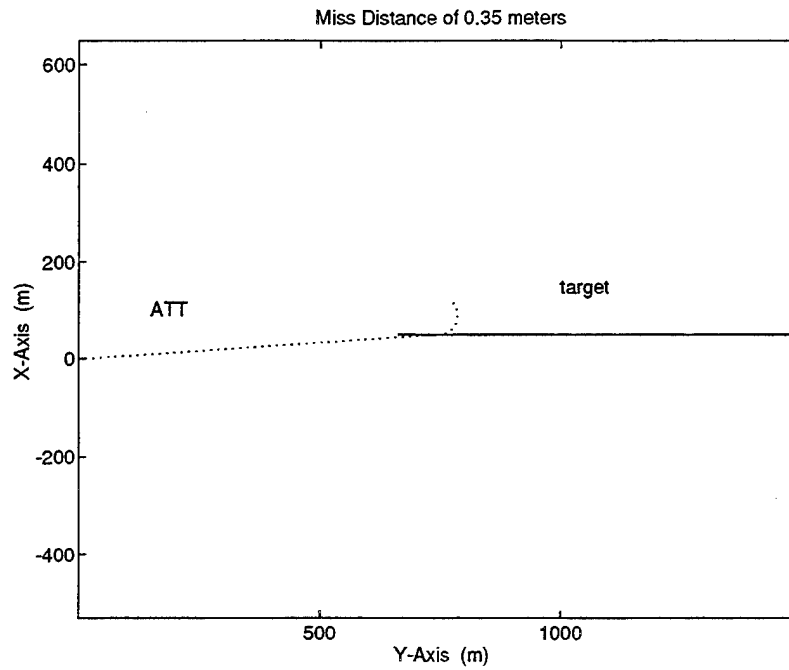


Figure 20. ATT and target trajectories with proportional navigation.

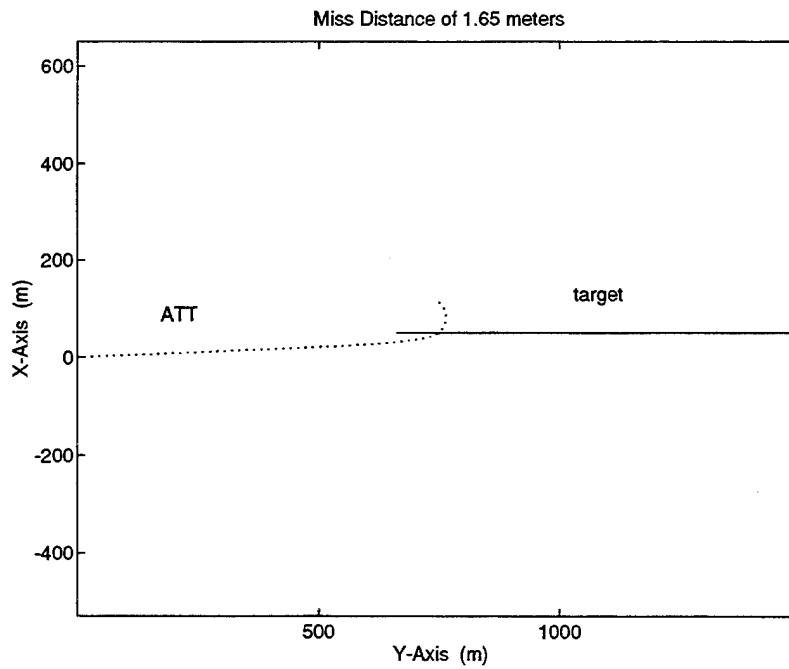


Figure 21. ATT and target trajectories with linear quadratic regulator.



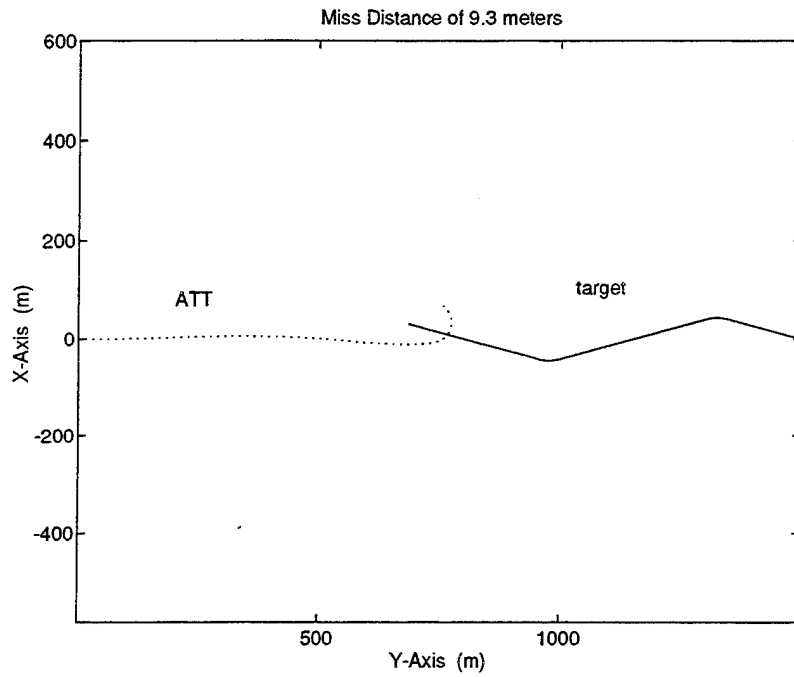


Figure 22. ATT and target trajectories with pursuit guidance.

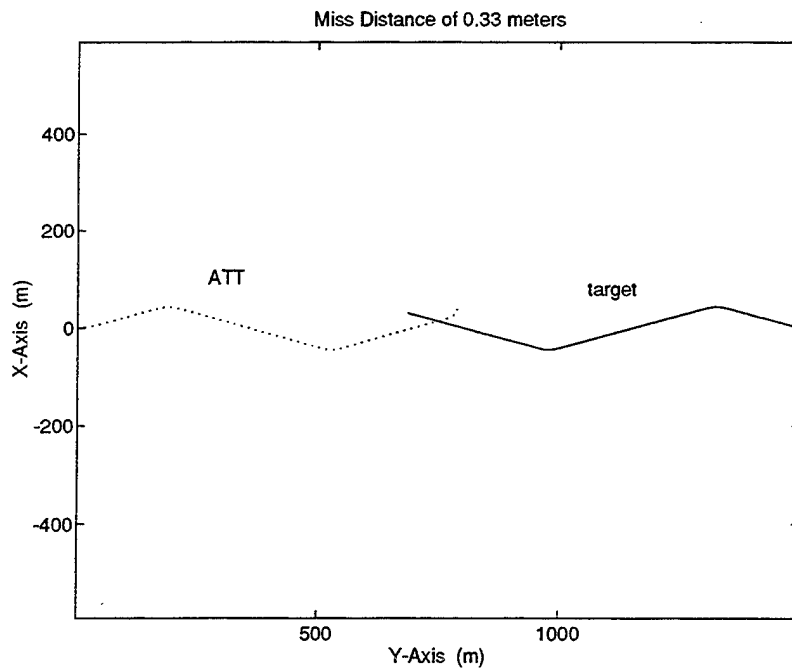


Figure 23. ATT and target trajectories with proportional navigation.

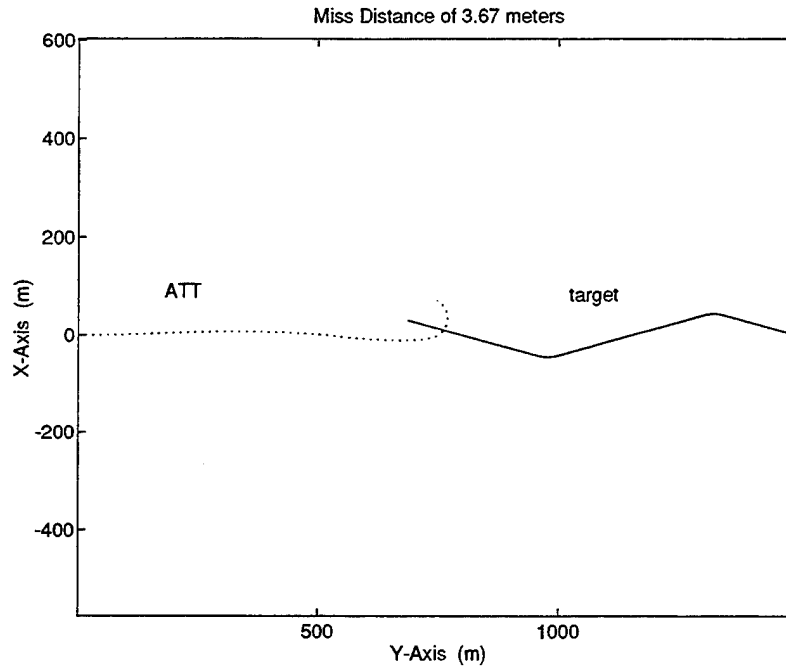


Figure 24. ATT and target trajectories with linear quadratic regulator.

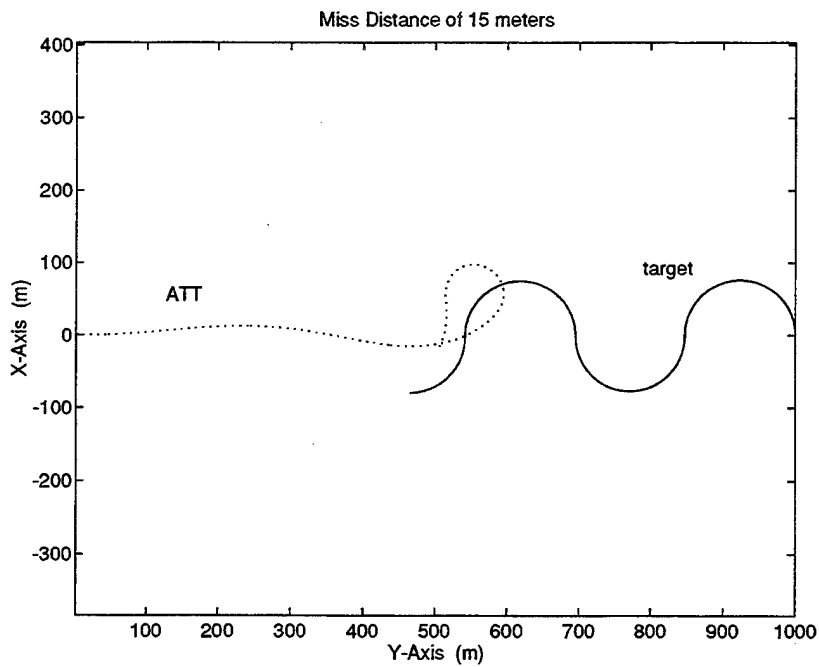


Figure 25. ATT and target trajectories with pursuit guidance.

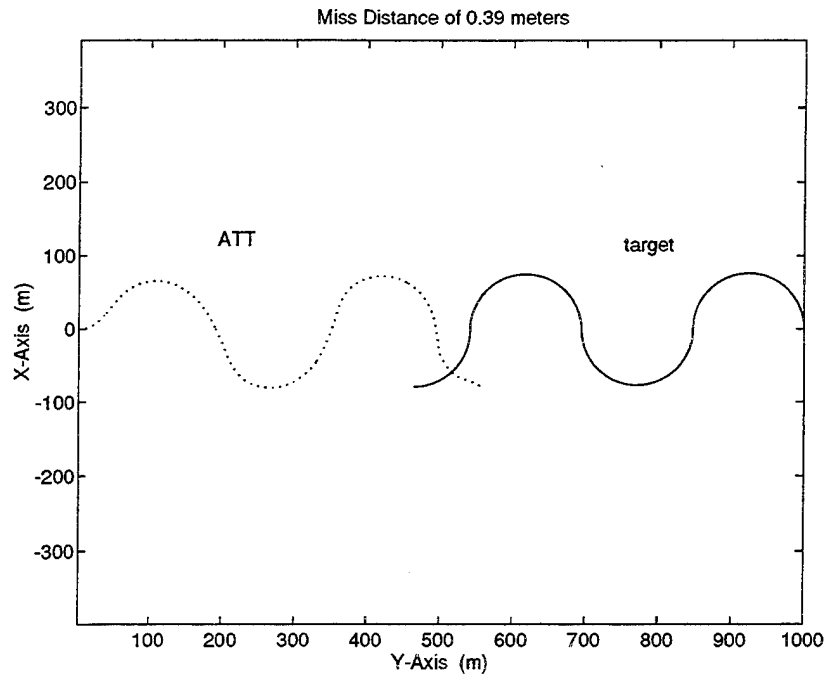


Figure 26. ATT and target trajectories with proportional navigation.

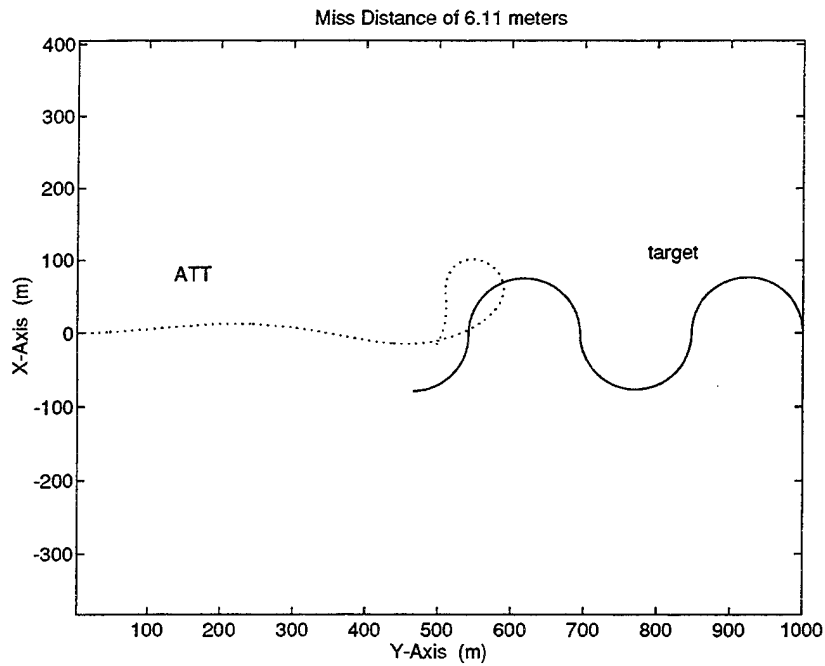


Figure 27. ATT and target trajectories with linear quadratic regulator.

The proportional navigation control algorithm is the one that produces the best results. The evaluation of the algorithm is based upon the miss distance. Even for the case where the target is performing a sinusoidal pattern, the miss distance is only 0.39 meters.

The analysis of the problem needs to take all factors into account. Simulations were made for noisy range and angle observations, and for noisy angle observations. Because of the poor Kalman filter estimates for the sinusoidal target maneuver, the miss distances were very large for all guidance laws, and results are not presented for that case (some simulations achieved miss distances of more than 50 meters). Also, because the linear quadratic regulator requires an estimate of all the system states, it was not used with angle only measurements. As seen in Chapter III, section E, the estimates are angle and angular rates instead of position and velocity.

For each different case, ten simulation runs were generated. Although ten runs cannot be considered large enough for the generation of important statistical conclusions, they are enough to evaluate the performance of the guidance laws. A summary of all results is shown on Tables 1, 2 and 3.

	RANGE & ANGLE		ANGLE		TRUE
	MEAN	STD	MEAN	STD	-
PURSUIT	4.33	0.16	4.16	0.11	3.78
PROPNAV	1.29	0.43	1.37	0.70	0.35
LINQUAD	2.03	0.18	-	-	1.65

Table 1. Miss Distances for a target in a straight line.

	RANGE & ANGLE		ANGLE		TRUE
	MEAN	STD	MEAN	STD	-
PURSUIT	10.51	0.17	9.8	0.20	9.30
PROPNAV	1.20	0.52	1.26	0.44	0.33
LINQUAD	5.36	0.34	-	-	3.67

Table 2. Miss Distances for a target in a zigzag pattern.

	RANGE & ANGLE		ANGLE		TRUE
	MEAN	STD	MEAN	STD	-
PURSUIT	-	-	-	-	15.00
PROPNAV	-	-	-	-	0.39
LINQUAD	-	-	-	-	6.11

Table 3. Miss Distances for a target in a sinusoidal pattern.

Again, the proportional navigation algorithm is the one with the best performance. In all the cases studied, the miss distances were small enough to predict a hit. The miss distance is evaluated between the centers of mass of both vehicles, and, miss distances of the order of the ones obtained correspond to hits. The linear quadratic regulator also has a small miss distance for a non-maneuvering target. It is important to remember that the poor results obtained for some algorithms were caused by the limitations imposed on the simulation, the similar speeds for both target torpedo and ATT, and the similar maneuverability characteristics. The maximum angular rate for the ATT was set to 0.5 radians per second ( approximately 30 degrees per second ).

## V. THE DOCKING PROBLEM

### A. GENERAL

This chapter describes the simulation of the docking problem and the control strategy found to solve it. The major difference between this problem and the previous one is that the objective is now to match position and velocity of both vehicles. Hence, this simulation addresses a harder part of the general docking problem: instead of docking to a fixed platform, the autonomous vehicle will dock to a moving target. The target, however, will move in a straight line at a constant speed. In order to solve the docking problem independently of the initial vehicle's position and velocity, the AUV has a speed advantage over the target. The limits on AUV maneuverability remain the same.

All the simulation procedures were modified to take into account the ability of the AUV to change speed. The guidance algorithm now calculates the desired angular rates and the desired vehicle speed. The equations of motion remain the same, but the AUV longitudinal speed is now an input. To simulate a first order drag term, the vehicle output speed is given by

$$\frac{y_{out}}{y_{desired}} = \frac{k}{s + k} \quad (5.1)$$

where  $k$  is a constant coefficient.

### B. DOCKING METHOD

The obvious guidance algorithm to be chosen, from the three mentioned in Chapter III, is the Linear Quadratic Regulator. This algorithm allows the minimization of the difference of both position and velocity between vehicles. The calculation of the control inputs is done separately for angular rates and speed. The angular rates are calculated according to equations 3.14 through 3.16, and the vehicle speed is another input. The minimization process is done by choosing the appropriate weighting matrices  $Q$  and  $R$ . However, the relative weights of position and velocity necessary to achieved a successful

docking are functions of the vehicles' relative positions and velocities. The way to solve it was to separate the process into three different stages.

The AUV initially closes to a position 10 meters astern of the target, aligned with the vehicles longitudinal axis and with a lateral separation of 20 meters. The lateral separation is chosen to be either to starboard or port of the target vehicle. During this initial approach, the ratio of position and velocity weights is equal to one. During this phase, the control inputs are calculated to move the AUV to the chosen position relative to the target and to align the AUV with the target longitudinal axis. This approach is done at a speed 2 m/s higher than the estimated target speed. The second step is to place the AUV at a position 5 meters alongside the target. The weights for position and velocity remain the same and the input speed is now equal to the estimated target speed plus 1 m/s. The third step is the final maneuver to accomplish docking. The position to velocity weight ratio is now 1/50, and the desired AUV speed is the estimated target speed plus 0.1 m/s. When the AUV is considered docked to the target, the AUV speed is reduced to the estimated target speed and the position to velocity weight ratio is changed to 1/500.

This docking process was simulated for a series of target initial positions and velocities. Figures 28 through 31 show several examples for different target initial conditions. In all the examples, two different AUV trajectories are shown, one for docking on each target side. The docking maneuver was effective in all cases for all target initial conditions and for both docking sides. The docking side is selected in the program that initializes the simulation process. Nevertheless, if the docking side is unimportant, the program can choose the one requiring the less control effort or the smaller time to docking.

Initial simulation trials used a single range/bearing sensor with rather coarse range accuracy, and results were poor. Later trials used the sensor accuracy described in section B of Chapter IV. For these simulation runs it was found that the performance of the control algorithm was about the same for true target information and for the Kalman estimates from the noisy range and angle measurements. This happens because we assumed that the AUV employed two different kinds of sensors, a long range one for the initial approach, and a more accurate one for the final approach.

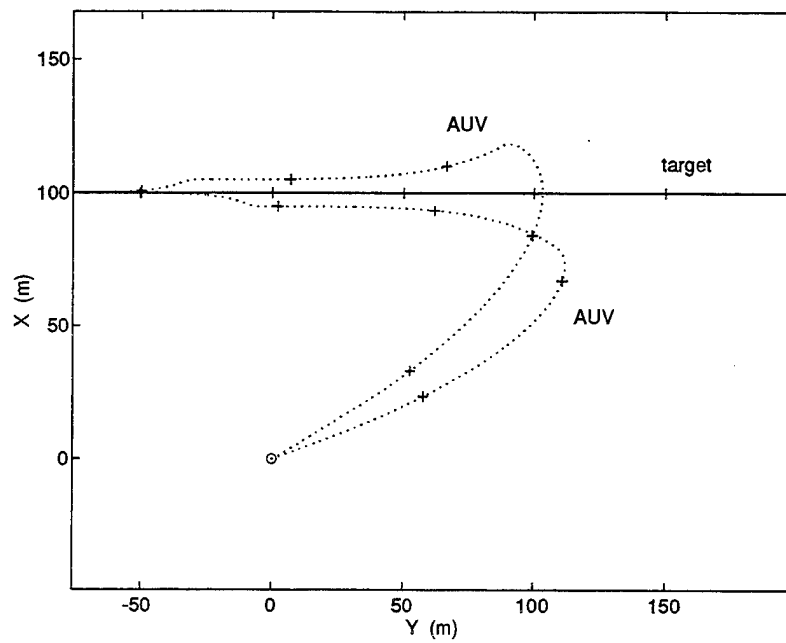


Figure 28. Docking maneuver, example 1.

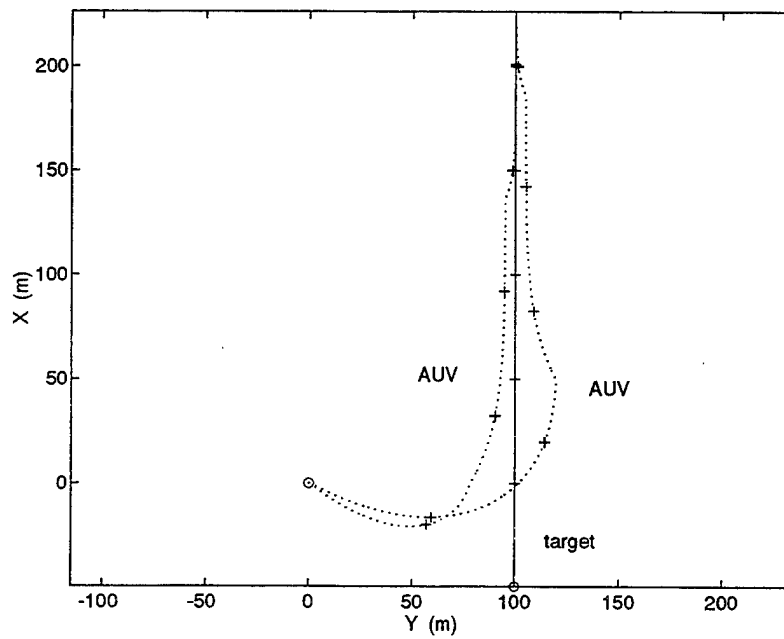


Figure 29. Docking maneuver, example 2.



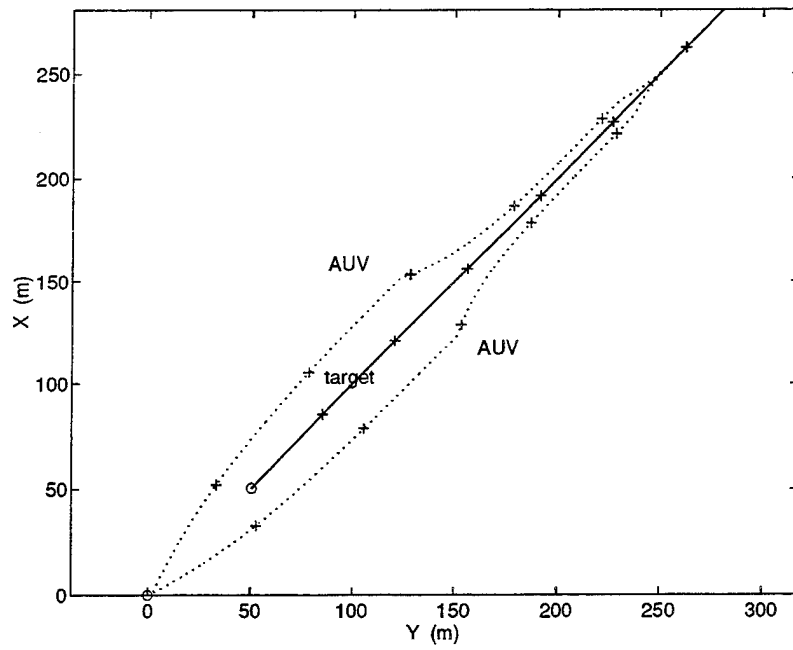


Figure 30. Docking maneuver, example 3.

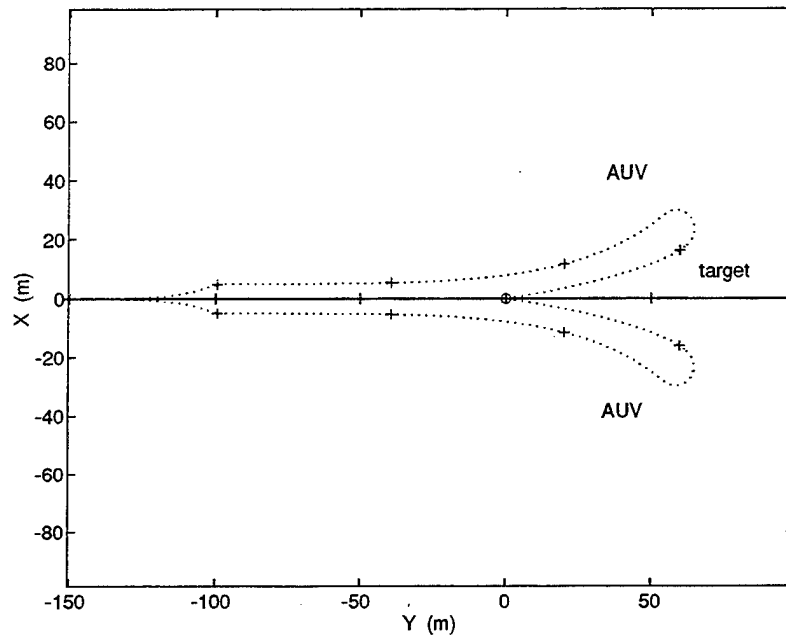


Figure 31. Docking maneuver, example 4.

## **VI. SUMMARY, CONCLUSIONS AND RECOMMENDATIONS**

This chapter summarizes all the simulation cases and presents thesis conclusions for sensors and guidance for the underwater intercept and docking problems. Some recommendations for follow-on work are also presented.

### **A. SUMMARY**

The interception and docking problems were studied using different assumptions. For the ATT interception problem, the target was allowed to follow three distinct trajectories: a straight line, a zigzag pattern, and a sinusoidal pattern. Three target information scenarios were used: perfect target information (both position and velocity), noisy range and angle measurements, and noisy angle measurements. Due to the poor velocity estimates for a target performing a sinusoidal pattern, this target trajectory was studied only when perfect target information was used. Three control laws: pursuit guidance, proportional navigation, and a linear quadratic regulator, were tested for the interception problem.

The docking problem was studied for a target moving in a straight line at constant speed, using both perfect target information and noisy range and angle measurements. The linear quadratic regulator was the control law used to maneuver the docking AUV.

### **B. CONCLUSIONS**

The thrust of this thesis was to evaluate the performance of the Kalman estimators and the control algorithms. The extended Kalman filter closely tracks target position and velocity for the straight line and zigzag cases. When angle only measurements are used, the line-of-sight angles are estimated with small errors. The angular rate estimates, because the pseudo observations are numerical derivatives of the noisy angle measurements, have a larger error. The Kalman filter does not produce a good estimate for target velocity for the sinusoidal case, and the angular rate estimates are also poor.

The delay in the reception of target measurement had a negligible effect on the miss distances obtained. If the sensors were placed at a location other than on the interceptor, this effect should be considered.

The proportional navigation algorithm achieves hits for all three target trajectories studied. Actually, the miss distances are approximately the same for all target trajectories. The linear quadratic regulator achieves reasonable miss distance for a target in a straight line, but its performance is poor in the other scenarios. The pursuit method is always bad. The reasons for the poor performance of the pursuit guidance are the speed and maneuverability restrictions imposed on the ATT.

The linear quadratic regulator, because of its ability to minimize velocity differences, is an adequate guidance law to solve the docking problem. The solution for the docking problem is always achieved, independently of target initial position and velocity, and docking side, as long as there exists a speed advantage of the AUV over the target.

### **C. RECOMMENDATIONS**

This study was done in a very general way. To be able to provide more answers on the feasibility of implementing the guidance algorithms on an actual vehicle, the dynamics of that specific vehicle should replace the general equations of motion used in this simulation. Once a real vehicle is simulated, the evaluation should be done in hit probability instead of average miss distance. For that, the calculation of the hit probability has to consider the dimensions of both target and interceptor vehicles.

The existence of other measurements, like doppler readings, was not considered in this thesis. Finally, to decrease the estimation errors when the target is turning, a target maneuver tracking package can be implemented.

## APPENDIX. MATLAB SOURCE CODE

The M-files and S-functions used in the simulation are presented in this appendix. For simplicity, the S-functions are presented as block diagrams.

### A. TRUE TARGET INFORMATION INTERCEPT

#### 1. System Initialization

```
clear
global Ts u time2 KF
A=[ 0 0 0 1 0 0 ;
    0 0 0 0 1 0 ;
    0 0 0 0 0 1 ;
    0 0 0 0 0 0 ;
    0 0 0 0 0 0 ;
    0 0 0 0 0 0 ];
C=eye(3,6);
B=[zeros(3);eye(3)];
D=zeros(6,3);% xdot(t)=Ax(t)+Bu(t) y(t)=Cx(t)+Du(t)
va=20;% att speed
vt=20;% target speed
% initial target Euler angles and position
%phi0=0;theta0=elevation;psi0=heading
%phi0t=0;theta0t=.05;psi0t=-pi/2;% linear
%xt0=50;yt0=1500;zt0=-100;
%phi0t=0;theta0t=.05;psi0t=-5*pi/12;% zigzag
%xt0=0;yt0=1500;zt0=-100;
phi0t=0;theta0t=.05;psi0t=0;% sinusoidal
xt0=0;yt0=1000;zt0=-100;%
%initial ATT Euler angles and position
phi0a=0;theta0a=0;psi0a=pi/2;
xa0=0;ya0=0;za0=-12;
Ts=.1;% sampling period
u=[0;0];% initial control input
time2=0;% Controller counter
tmax=42;% final time
```

## 2. Simulation Diagram

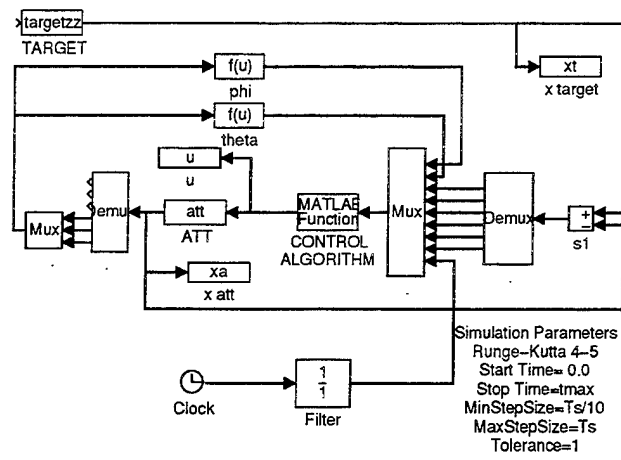


Figure 32. True target information intercept simulation diagram.

### 3. Target Dynamics

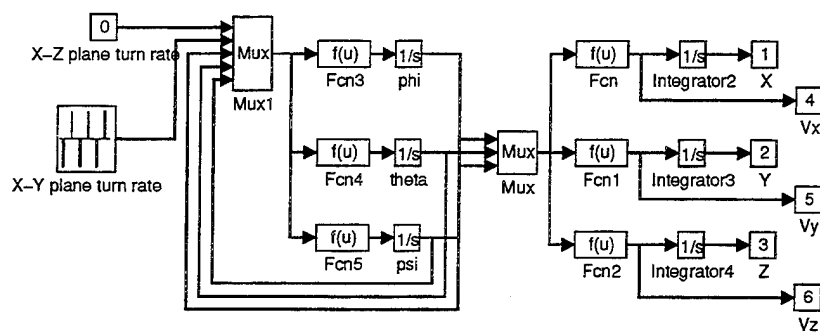


Figure 33. Target dynamics.

#### 4. ATT Dynamics

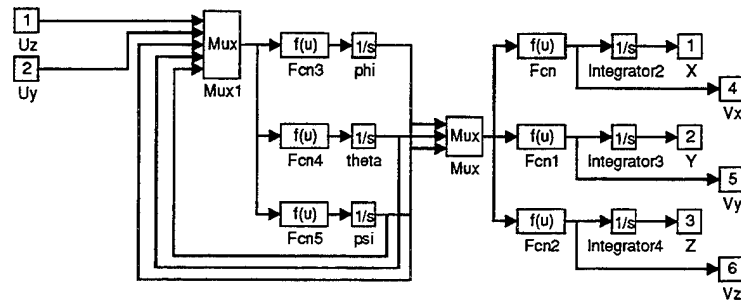


Figure 34. ATT dynamics.

#### 5. Pursuit Guidance

```
function y=pursuit(x)
% PURSUIT GUIDANCE ALGORITHM
%
% input: [      ATT heading and angle of site      x(1:2);
%          relative target position and velocities in absolute coordinates x(3:8);
%          time                                       x(9); ]
%
% output: [      command accelerations in body centered coordinates y(1:2); ]
%
global u Ts time2
% calculations made iff [ time = n * Ts ] n=1,2,3,...
if x(9)~=time2
    time2=x(9);
    fi=x(1);
    teta=x(2);
    % rotation matrix
    Lbi=[ cos(fi)*cos(teta) sin(fi)*cos(teta) -sin(teta) ;
          -sin(fi)         cos(fi)         0 ;
          cos(fi)*sin(teta) sin(fi)*sin(teta) cos(teta) ];
    % rotate position to body centered coordinates
    rb=Lbi*[x(3);x(4);x(5)];
```

```

    alfay=atan2(rb(2),sqrt(rb(1)^2+rb(2)^2+rb(3)^2));
    alfaz=atan2(rb(3),sqrt(rb(1)^2+rb(2)^2+rb(3)^2));
    kp=3;
    uay=kp*alfay;
    uaz=kp*alfaz;
    umax=.5;
    utot=sqrt(uay^2+uaz^2);
    if utot>umax
        uay=uay*umax/utot;
        uaz=uaz*umax/utot;
    end
    u=[uaz;uay];
end
y=u;

```

## 6. Proportional Navigation

```

function y=propnav(x)
% PROPORTIONAL NAVIGATION ALGORITHM
%
% input: [      ATT heading and angle of site      x(1:2);
%          relative target position and velocities in absolute coordinates x(3:8);
%          time                                       x(9);   ]
%
% output: [      command accelerations in body centered coordinates  y(1:2);  ]
%
global u Ts time2
% calculations made iff [ time = n * Ts ] n=1,2,3,...
%
if x(9)~=time2
    time2=x(9);
    fi=x(1);
    teta=x(2);
    % rotation matrix
    Lbi=[ cos(fi)*cos(teta) sin(fi)*cos(teta) -sin(teta)      ;
          -sin(fi)         cos(fi)           0              ;
          cos(fi)*sin(teta) sin(fi)*sin(teta) cos(teta) ];
    % rotate position and velocity to body centered coordinates
    rb=Lbi*[x(3);x(4);x(5)];
    vb=Lbi*[x(6);x(7);x(8)];
    wb=cross(rb,vb)/norm(rb)^2;
    ex=[1;0;0];
    int=cross(wb,ex);
    ep=int/norm(int);
    kpn=4;
    uay=kpn*ep(2)*norm(wb)*norm(vb)/Ts/va;
    uaz=kpn*ep(3)*norm(wb)*norm(vb)/Ts/va;
    umax=0.5;
    utot=sqrt(uay^2+uaz^2);
    if utot>umax

```

```

        uay=uay*umax/utot;
        uaz=uaz*umax/utot;
    end
    u=[uaz;uay];
end
y=u;

```

## 7. Linear Quadratic Regulator

```

function y=linquad(x)
% LINEAR QUADRATIC REGULATOR ALGORITHM
%
% input: [      ATT heading and angle of site      x(1:2);
%          relative target position and velocities in absolute coordinates x(3:8);
%          time                                       x(9);
%
% output: [      command accelerations in body centered coordinates  y(1:2); ]
%
global u Ts time2 KF
% calculations made iff [ time = n * Ts ] n=1,2,3,...
%
if x(9)~=time2
    time2=x(9);
    fi=x(1);
    teta=x(2);
    % rotation matrix
    Lbi=[ cos(fi)*cos(teta) sin(fi)*cos(teta) -sin(teta)      ;
          -sin(fi)         cos(fi)         0                ;
          cos(fi)*sin(teta) sin(fi)*sin(teta) cos(teta) ];
    % rotate position and velocity to body centered coordinates
    rb=Lbi*[x(3);x(4);x(5)];
    vb=Lbi*[x(6);x(7);x(8)];
    ua=KF*[rb;vb];
    uax=ua(1)/va;
    uay=ua(2)/va;
    uaz=ua(3)/va;
    umax=.5;
    utot=sqrt(uay^2+uaz^2);
    if utot>umax
        uay=uay*umax/utot;
        uaz=uaz*umax/utot;
    end
    u=[uaz;uay];
end
y=u;

```



## B. RANGE AND ANGLE MEASUREMENTS INTERCEPT

### 1. System Initialization

```

clear
global Ts F C R I H Q P xhat u time1 time2 KF
A=[    0 0 0 1 0 0 ;
      0 0 0 0 1 0 ;
      0 0 0 0 0 1 ;
      0 0 0 0 0 0 ;
      0 0 0 0 0 0 ;
      0 0 0 0 0 0];
C=eye(3,6);
B=[zeros(3);eye(3)];
D=zeros(6,3);% xdot(t)=Ax(t)+Bu(t) y(t)=Cx(t)+Du(t)
va=20;% att speed
vt=20;% target speed
% initial target Euler angles and position
%phi0=0;theta0=elevation;psi0=heading
%phi0t=0;theta0t=.05;psi0t=-pi/2;% linear
%xt0=50;yt0=1500;zt0=-100;
%phi0t=0;theta0t=.05;psi0t=-5*pi/12;% zigzag
%xt0=0;yt0=1500;zt0=-100;
phi0t=0;theta0t=.05;psi0t=0;% sinusoidal
xt0=0;yt0=1000;zt0=-100;%
%initial ATT Euler angles and position
phi0a=0;theta0a=0;psi0a=pi/2;
xa0=0;ya0=0;za0=-12;
I=eye(6);
Ts=1;% sampling period
[F,H]=c2d(A,B,Ts);% x(k+1)=F(ts)x(k)+H(Ts)u(k)
QQ=500*[eye(3,6);zeros(3,6)];% Cost=integral (xQQx+uRRu)dt
RR=eye(3);
KF=dlqr(F,H,QQ,RR);% feedback gain
root=0;% white noise seed
sr=10;sa=1*(pi/180);sb=sa;
R=[sr^2 0 0;0 sa^2 0;0 0 sb^2];% measurement noise covariance matrix
q=1;
SIG=[q^2*Ts^3/3 q^2*Ts^2/2;q^2*Ts^2/2 q^2*Ts];
O=zeros(2);
Q=[SIG O O;O SIG O;O O SIG];% discrete plant noise covariance matrix
P=1e5*I;% initial state error covariance matrix
xhat=[xt0;yt0;zt0;0;-vt;0];% initial Kalman estimate
u=[0;0];% initial control input
time1=0;% Kalmal filter counter
time2=0;% Controller counter
tmax=42;% final time

```

## 2. Simulation Diagram

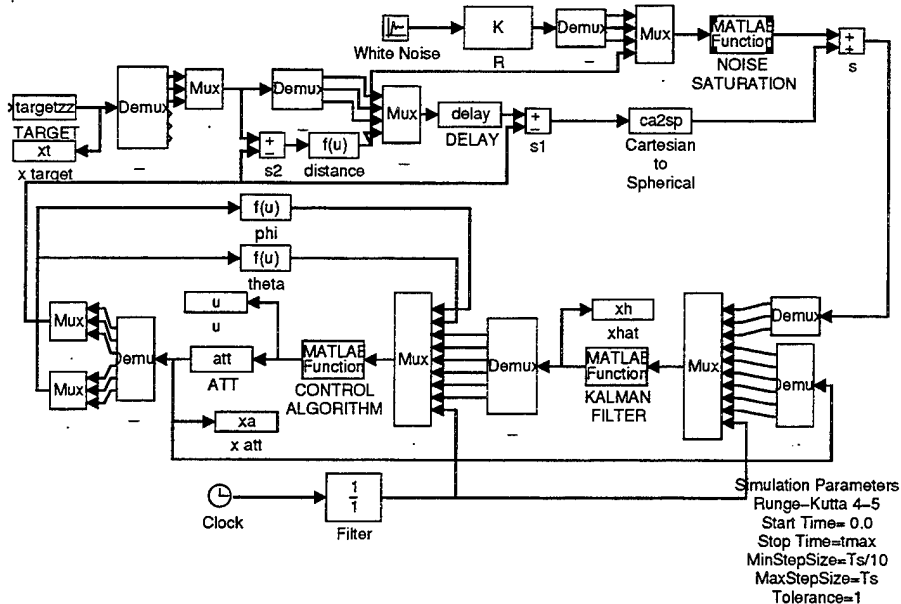


Figure 35. Range and angle measurements intercept simulation diagram.

### 3. Observations Delay

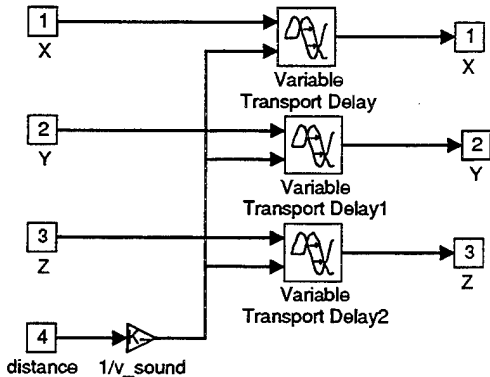


Figure 36. Observations delay.

#### 4. Noise Saturation

```
function y=maxnoise(x)
% NOISE REGULATOR
%
% input: [      Noise                x(1:3);
%          Target distance from ATT    x(4);  ]
%
% output: [      min(x(1),x(4)/10);x(2);x(3)      ]
%
if (abs(x(1))>(x(4)/10))
    dr=x(4)/10*sign(x(1));
else
    dr=x(1);
end
y=[dr;x(2);x(3)];
```

#### 5. Extended Kalman Filter

```
function y=ekf_pseu(x)
% DISCRETE TIME EXTENDED KALMAN FILTER
%
% input: [      measurements    x(1:3) ; (range + bearing + elevation)
%          ATT position         x(4:6) ;
%          ATT velocity         x(7:9) ;
%          time                  x(10);      ]
%
% output: [      relative estimated target position and velocities    y(1:6);  ]
%
global Ts F C R I Q P xhat time1
xa=x(4:9);% ATT position and velocity
% calculations made iff [ time = n * Ts ] n=1,2,3,...
%
if x(10)~=time1
    time1=x(10);
    xhat=F*xhat;
    P=F*P*F'+Q;
    r=x(1);a=x(2);b=x(3);
    rho=tan(a)^2*tan(b)^2+tan(a)^2+tan(b)^2+1;
    ta=tan(a);tb=tan(b)^2;
    ta12=ta^2+1;tb12=tb^2+1;
    G=[    1/sqrt(rho)      -r*ta*ta12*tb12/rho^(3/2)  -r*tb*ta12*tb12/rho^(3/2) ;
        ta/sqrt(rho)      r*ta12*tb12/rho^(3/2)      r*ta*tb*ta12*tb12/rho^(3/2);
        tb/sqrt(tb12)      0                          r/sqrt(tb12)                ];
    rhat=norm(xhat(1:3)-x(4:6));
    if rhat<100
        R(1,1)=1;
    end
end
```

```

Rstar=G*R*G';
K=P*C'*inv(C*P*C'+Rstar);
[z(1),z(2),z(3)]=sph2cart(a,b,r);
yhat=C*(xhat-xa);
xhat=xhat+K*(z'-yhat);
P=(I-K*C)*P*(I-K*C)+K*Rstar*K';
end
y=[xhat-xa];

```

## C. ANGLE ONLY MEASUREMENTS INTERCEPT

### 1. System Initialization

```

clear
global Ts F C R I Q P xhat u time1 time2
A=[ 0 0 1 0 ;
    0 0 0 1 ;
    0 0 0 0 ;
    0 0 0 0 ];
C=eye(4);
B=[zeros(2);eye(2)];
D=zeros(4,2);% xdot(t)=Ax(t)+Bu(t) y(t)=Cx(t)+Du(t)
va=20;% att speed
vt=20;% target speed
% initial target Euler angles and position
%phi0=0;theta0=elevation;psi0=heading
%phi0t=0;theta0t=.05;psi0t=-pi/2;% linear
%xt0=50;yt0=1500;zt0=-100;
%phi0t=0;theta0t=.05;psi0t=-5*pi/12;% zigzag
%xt0=0;yt0=1500;zt0=-100;
phi0t=0;theta0t=.05;psi0t=0;% sinusoidal
xt0=0;yt0=1000;zt0=-100;%
%initial ATT Euler angles and position
phi0a=0;theta0a=0;psi0a=pi/2;
xa0=0;ya0=0;za0=-12;
I=eye(4);
Ts=.1;% sampling period
[F,H]=c2d(A,B,Ts);% x(k+1)=F(ts)x(k)+H(Ts)u(k)
root=0;% white noise seed
sa=1*(pi/180);sb=sa;
R=[sa^2 0 0 0;0 sb^2 0 0;0 0 5 0;0 0 0 5];% measurement noise covariance matrix
q=1;
SIG=[q^2*Ts^3/3 q^2*Ts^2/2;q^2*Ts^2/2 q^2*Ts];
O=zeros(2);
Q=[SIG O;O SIG];% discrete plant noise covariance matrix
P=1e5*I;% initial state error covariance matrix
xhat=[atan2(-xt0,yt0);atan2(-zt0,sqrt(yt0^2+xt0^2));0;0];% initial Kalman estimate
u=[0;0];% initial control input
time1=0;% Kalman filter counter

```

```
time2=0;% Controller counter
tmax=42;% final time
```

## 2. Simulation Diagram

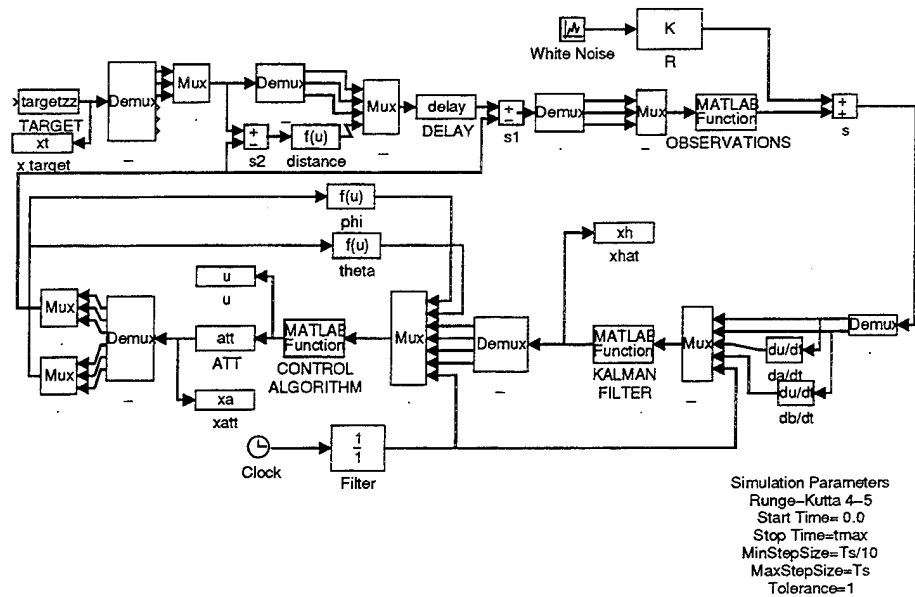


Figure 37. Angle only measurements intercept simulation diagram.

## 3. Kalman Filter

```
function y=ekf_ao(x)
% DISCRETE TIME EXTENDED KALMAN FILTER
%
% input: [      measurements      x(1:2); (bearing + elevation)
%          x(3:4); (pseudo angular rates)
%          time                    x(5);
%
% output: [      relative estimated target angles and angular rates      y(1:2); ]
%
global Ts F R I C Q P xhat time1
% calculations made iff [ time = n * Ts ] n=1,2,3,...
%
if x(5)~=time1
    time1=x(5);
    xhat=F*xhat;
    P=F*P*F'+Q;
```

```

K=P*C'*inv(C*P*C'+R);
diff=x(1:4)-C*xhat;
diff=asin(sin(diff));
xhat=xhat+K*diff;
P=(I-K*C)*P*(I-K*C)+K*R*K';
end
y=[xhat];

```

#### 4. Pursuit Guidance

```

function y=pursuit(x)
% PURSUIT GUIDANCE ALGORITHM ANGLE ONLY
%
% input: [      Observed bearing, elevation and rates of change      (3:6);
%          time                                           x(7);   ]
%
% output: [      command accelerations in body centered coordinates  y(1:2);   ]
%
global u Ts time2
% calculations made iff [ time = n * Ts ] n=1,2,3,...
%
if x(7)~=time2
    time2=x(7);
    f=x(1);t=x(2);a=x(3);b=x(4);
    px=cos(b)*cos(a);py=cos(b)*sin(a);pz=sin(b);
    alfay=atan2(-sin(f)*px+cos(f)*py,cos(f)*sin(t)*px+sin(f)*cos(f)*py-sin(t)*pz);
    alfaz=atan2(cos(f)*sin(t)*px+sin(f)*sin(t)*py+cos(t)*pz,cos(f)*sin(t)*px+
        sin(f)*cos(f)*py-sin(t)*pz);
    kp=3;
    uay=kp*alfay;
    uaz=kp*alfaz;
    umax=0.5;
    utot=sqrt(uay^2+uaz^2);
    if utot>umax
        uay=uay*umax/utot;
        uaz=uaz*umax/utot;
    end
    u=[uaz,uay];
end
y=u;

```

#### 5. Proportional Navigation

```

function y=propn_ao(x)
% PROPORTIONAL NAVIGATION ALGORITHM ANGLE ONLY
%
% input: [      Observed bearing, elevation and rates of change      x(3:6);
%          time                                           x(7);   ]

```

```

%
% output: [      command accelerations in body centered coordinates   y(1:2);  ]
%
global u Ts time2
% calculations made iff [ time = n * Ts ] n=1,2,3,...
%
if x(7)~=time2
    time2=x(7);
    kpn=4;
    uay=x(5)*kpn;
    uaz=x(6)*kpn;
    umax=0.5;
    utot=sqrt(uay^2+uaz^2);
    if utot>umax
        uay=uay*umax/utot;
        uaz=uaz*umax/utot;
    end
    u=[uaz;uay];
end
y=u;

```

## D. DOCKING PROBLEM

### 1. System Initialization

```

clear
global Ts F R I H Q P xhat u time1 time2 KF0 KF1 KF2 vt vm dA dB1 dB2 flag sr sa sb
A=[ 0 0 0 1 0 0 ;
    0 0 0 0 1 0 ;
    0 0 0 0 0 1 ;
    0 0 0 0 0 0 ;
    0 0 0 0 0 0 ;
    0 0 0 0 0 0];
C=eye(6);
B=[zeros(3);eye(3)];
D=zeros(6,3);% xdot(t)=Ax(t)+Bu(t) y(t)=Cx(t)+Du(t)
vm=8;% max. ATT speed
vt=5;% target speed
%phi0=0;theta0=elevation;psi0=heading
%phi0t=0;theta0t=0;psi0t=-pi/2;    %      #1
%xt0=100;yt0=200;zt0=-12;
%phi0t=0;theta0t=0;psi0t=0;        %      #2
%xt0=-50;yt0=100;zt0=-12;
%phi0t=0;theta0t=0;psi0t=pi/4;    %      #3
%xt0=50;yt0=50;zt0=-12;
phi0t=0;theta0t=0;psi0t=-pi/2;    %      #4
xt0=0;yt0=100;zt0=-12;
phi0a=0;theta0a=0;psi0a=pi/2;% initial ATT Euler angles
xa0=0;ya0=0;za0=-12;% initial ATT positions

```

```

I=eye(6);
Ts=.1;% sampling period
[F,H]=c2d(A,B,Ts);%  $x(k+1)=F(ts)x(k)+H(Ts)u(k)$ 
QQ0=500*eye(6);%  $Cost=\int (xQQx+uRRu)dt$ 
QQ1=[100*eye(3,6);zeros(3) 500*eye(3)];
QQ2=[eye(3,6);zeros(3) 500*eye(3)];
RR=0.1*eye(3);
KF0=dlqr(F,H,QQ0,RR);% feedback gain
KF1=dlqr(F,H,QQ1,RR);
KF2=dlqr(F,H,QQ2,RR);
root=0;% white noise seed
sr=5;sa=1*(pi/180);sb=sa;
R=[sr^2 0 0;0 sa^2 0;0 0 sb^2];% measurement noise covariance matrix
q=2;SIG=[q^2*Ts^3/3 q^2*Ts^2/2;q^2*Ts^2/2 q^2*Ts];
O=zeros(2);
Q=[SIG O O;O SIG O;O O SIG];% discrete plant noise covariance matrix
P=1e5*I;% initial state error covariance matrix
xhat=[xt0;yt0;zt0;0;-vt;0];% initial Kalman estimate
u=[0;0;0];% initial control input
time1=0;% Kalman filter counter
time2=0;% Controller counter
tmax=50;% final time
dA=20;% distance behind
dB1=-20;% 1st distance alongside
dB2=-5;% 2nd distance alongside
flag=0;% control for set point

```



## 2. Simulation Diagram

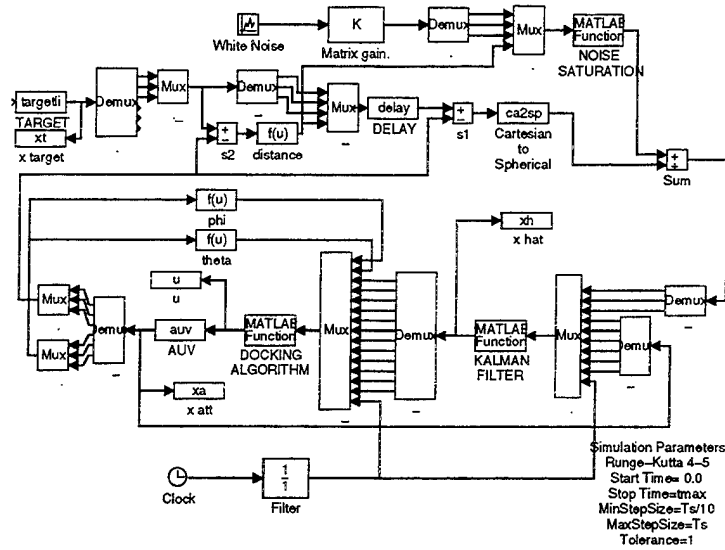


Figure 38. Docking simulation diagram.

## 3. Docking Algorithm

```
function y=dock(x)
%
% DOCKING ALGORITHM
%
% input: [      AUV heading and angle of site      x(1:2);
%             relative target position and velocities in absolute coordinates x(3:8);
%             time                                     x(9);
%
% output: [      command accelerations in body centered coordinates y(1:2);
%             Desired AUV speed                                     y(3); ]
%
global u Ts time2 KF0 KF1 KF2 vm vt dB1 dB2 dA flag
if x(15)~=time2
    time2=x(15);
    fi=x(1);
    teta=x(2);
    Lbi=[ cos(fi)*cos(teta) sin(fi)*cos(teta) -sin(teta) ;
          -sin(fi)         cos(fi)         0          ;
          cos(fi)*sin(teta) sin(fi)*sin(teta) cos(teta) ];
    if flag==0
        distT=[x(6)/vt*dA;x(7)/vt*dA;0]+[x(7)/vt*dB1;-x(6)/vt*dB1;0];
        va=vt+2;
```

```

elseif flag==1
    distT=[x(7)/vt*dB2;-x(6)/vt*dB2;0];
    va=vt+1;
elseif flag==2
    distT=0;
    va=vt+.1;
else
    distT=0;
    va=vt;
end
rbr=[x(3);x(4);x(5)]-[x(9);x(10);x(11)];
vbr=[x(6);x(7);x(8)]-[x(12);x(13);x(14)];
if (norm(rbr-distT)<3 & flag==0)
    flag=1;
elseif (norm(rbr-distT)<1 & flag==1)
    flag=2;
elseif (norm(rbr)<.2 & flag==2)
    flag=3;
end
if (flag==0 | flag==1)
    KF=KF0;
elseif flag==2
    KF=KF1;
else
    KF=KF2;
end
rb=Lbi*(rbr-distT);
vb=Lbi*vbr;
ua=KF*[rb;vb];
uax=ua(1);
uay=ua(2);
uaz=ua(3);
umax=20;
utot=sqrt(uay^2+uaz^2);
if utot>umax
    uay=uay*umax/utot;
    uaz=uaz*umax/utot;
end
u=[uaz;uay;va];
end
y=u;

```



## LIST OF REFERENCES

1. R. J. Boncal, " A Study of Based Maneuvering Controls for Autonomous Underwater Vehicles," Mechanical Engineer's Thesis, Naval Postgraduate School, Monterey, CA, 1987.
2. G. M. Anderson, " Guidance Algorithms for Anti-Torpedo Torpedos," Final Report, Naval Coastal Systems Center, Panama City, FL, 1989.
3. D. E. Kirk, *Optimal Control Theory, An Introduction*, Prentice-Hall, Englewood Cliffs, NJ, 1970.
4. A. Gelb (ed), *Applied Optimal Estimation*, The MIT Press, Cambridge, MA, 1974.
5. J. H. Blakelock, *Automatic Control of Aircraft and Missiles*, 2<sup>nd</sup> edition, New York, NY, Wiley Interscience, 1991.
6. C. F. Lin, *Modern Navigation, Guidance, and Control Processing*, Prentice-Hall, Englewood Cliffs, NJ, 1991.



## INITIAL DISTRIBUTION LIST

		No. of Copies
1.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2.	Library Code 52 Naval Postgraduate School Monterey, CA 93943-5101	2
3.	Chairman, Code EC Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943-5121	1
4.	Professor Robert G. Hutchins, Code EC/Hu Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943-5121	1
5.	Professor Roberto Cristi, Code EC/Cx Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943-5121	1
6.	Direcção do Serviço de Instrução e Treino Administração Central de Marinha Rua do Arsenal 1100 Lisboa Portugal	1
7.	LTJG João P. C. Roque Vivenda Grilo Farinheiras, Paio Pires 2840 Seixal Portugal	1